Semiannual Technical Summary

Information Processing Techniques Program

Volume I: Packet Speech Systems Technology

31 March 1980

Prepared for the Defense Advanced Research Projects Agency
under Electronic Systems Division Contract F19628-80-C-0002 by

Lincoln Laboratory

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

LEXINGTON, MASSACHUSETTS

The work reported in this document was performed at Lincoln Laboratory, a center for research operated by Massachusetts Institute of Technology. This work was sponsored by the Defense Advanced Research Projects Agency under Air Force Contract F19628-80-C-0002 (ARPA Order 3673).

This report may be reproduced to satisfy needs of U.S. Government agencies.

This technical report has been reviewed and is approved for publication.

FOR THE COMMANDER

Raymond L. Loiselle

Raymond L. Loiselle, Lt. Col., USAF
Chief, ESD Lincoln Laboratory Project Office

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

LINCOLN LABORATORY

INFORMATION PROCESSING TECHNIQUES PROGRAM

VOLUME I: PACKET SPEECH SYSTEMS TECHNOLOGY

SEMIANNUAL TECHNICAL SUMMARY REPORT
TO THE
DEFENSE ADVANCED RESEARCH PROJECTS AGENCY

1 OCTOBER 1979 – 31 MARCH 1980

ISSUED 24 SEPTEMBER 1980

LEXINGTON MASSACHUSETTS

ABSTRACT

This report describes work performed on the Packet Speech Systems Technology Program sponsored by the Information Processing Techniques Office of the Defense Advanced Research Projects Agency during the period 1 October 1979 through 31 March 1980.

CONTENTS

INTRODUCTION AND SUMMARY

The long-range objectives of the Packet Speech Systems Technology Program are to develop and demonstrate techniques for efficient digital speech communication on networks suitable for both voice and data, and to investigate and develop techniques for integrated voice and data communication in packetized networks, including wideband common-user satellite links. Specific areas of concern are: the concentration of statistically fluctuating volumes of voice traffic; the adaptation of communication strategies to conditions of jamming, fading, and traffic volume; and the eventual interconnecting of wideband satellite networks to terrestrial systems.

Previous efforts in this area have led to new vocoder structures for improved narrowband voice performance and multiple-rate transmission, and to demonstrations of conversational speech and conferencing on the ARPANET and the Atlantic Packet Satellite Network.

The current program has two major thrusts, i.e., the development and refinement of practical low-cost, robust, narrowband and variable-rate speech algorithms and voice terminal structures, and the establishment of an experimental wideband satellite network to serve as a unique facility for the realistic investigation of voice/data networking strategies. Efforts prior to FY 79 in these two areas were reported separately in the Packet Speech and Wideband Integrated Voice/Data Technology Program Semiannual Technical Summaries.

This report covers work in the following areas: development of a custom-LSI channel vocoder; hardware implementation study for the Spectral Envelope Estimation (SEE) vocoder; studies and real-time implementations of improved filter-bank vocoder structures emphasizing high-quality, multiple-transmission rates and integrated acoustic-noise stripping; design and implementation of compact, modular packet voice terminals and local-area-access facilities; the development of advanced voice protocols to be evaluated in the wideband satellite experiments; the design and implementation of a miniconcentrator facility to mediate the flow of local-access-area traffic onto a wideband satellite channel; wideband experiment definition and planning status; progress in satellite network and internetted packet voice conferencing; a design study of an ultrahigh performance programmable research-oriented speech processor; and narrowband speech consortium support.

Hardware implementation studies for the SEEVOC (SEE vocoder) indicate that an all-digital approach based on a mini-array processor is the most promising approach. A high-quality, 29-channel dual-rate vocoder has been implemented and evaluated in a non-real-time simulation. An integrated acoustic-noise-stripping feature has been incorporated into the earlier 19-channel realization. Prototypes of the first-generation packet voice terminal/LEXNET equipments have been successfully demonstrated in a point-to-point call mode over 1000 ft of cable using a rudimentary dial-up/ring protocol. An advanced version of the packet voice

processor has been defined and is in the final stages of detailed design. First-generation definition of NVP-II and ST have been carried out, with detailed implementation and further refinement currently in progress. The hardware and software designs for a miniconcentrator facility are complete, and detailed implementation is under way. Supplement 1 to the original Wideband Experiment Plan has been completed and is being published as a separate document. The hardware design study for an F100K technology-based advanced speech processor is under way, and several architectural alternatives to optimize the cost/complexity/performance trade-off are being considered.

PACKET SPEECH SYSTEMS TECHNOLOGY

I. CCD/BELGARD CHANNEL VOCODER

Texas Instruments delivered several partially functional synthesizer devices for the NMOS LSI channel vocoder. The designers had identified a number of problem areas and have taken corrective steps which will be implemented in the next processing iteration. Known anomalies include faulty excitation subsystems, low Q's in the resonator filter bank, improper switching of the two channel 19 filters, and an overall poor signal-to-noise ratio (S/N). A stand-alone test jig exclusive of the vocoder prototype was fabricated to operate the devices. The test jig derived the necessary operating voltages, supplied output analog audio, and incorporated the necessary interface hardware for a Lincoln Digital Signal Processor (LDSP) connection. The LDSP testbed software was modified to conform to the revised in/out protocols, and the devices were successfully exercised. The analysis and pitch extraction functions were included in the real-time software. A real-time simulated synthesizer output was also provided for direct performance comparison with the actual chip. A substantial amount of high-quality source material (dynamic microphone) was processed and recorded along with the simulation output for reference. Aside from the expected degradations attributable to poor S/N and the other design flaws previously noted, some further quality anomalies were noticed whose origins were not immediately apparent. The output recordings were forwarded to Texas Instruments for detailed analysis.

II. VOCODER STUDIES

A. Spectral Envelope Estimation Vocoder (SEEVOC)

Several short studies relating to the SEEVOC have been performed: two critical band-based samplings of the log spectral envelope were tested, an inefficiency due to DPCM coding has been estimated, a published scheme for applying phase to the synthesizer has also been tried, the tandeming properties of the SEE algorithm have been investigated, and an implementation study of the algorithm has been performed.

The current SEE encoding scheme is based on a uniform sampling (every 90 Hz) of the spectral envelope. Auditory perception theory suggests that the higher-frequency regions of the spectrum need not be sampled as closely as the low-frequency portions. Therefore, two new coding schemes based on critical bands were designed (see Table I). Both involve modified samplings of

TABLE I
SEE ENCODING SCHEMES

	Low Frequency		High Frequency		
	Zone (Hz)	Interval (Hz)	Zone (Hz)	Spacing Factor	Total Samples
Version 1	0 to 540	90	540 to 3788	1.2	17
Version 2	0 to 990	90	990 to 3788	1.1	27

the current 42-point spectral envelope waveforms currently in use. The spacing factor is the ratio of each zonal center frequency to the preceding one in the high-frequency region. The first corresponds to a one-sample-per-critical-band approach, and the second is a two-sample-per-critical-band version. Several methods for evaluating the samples and regenerating the 42-point spectral envelope were tried. All resulted in a muffled distortion of the speech, suggesting that either the critical-band-based samplings or the methods employed for sampling and desampling were not appropriate for representing the speech spectral envelope.

The DPCM scheme used for encoding the log spectral envelope is capable of describing many spectra which cannot be produced by the human voice or perceived by the human ear. Since DPCM limits the slope of a waveform but not its magnitude, the dynamic range which can be described for the last (highest) frequency sample is over 400 dB. Given that a much more reasonable value for the dynamic range of speech is 90 dB, a count was made of all possible spectral descriptions which remained within this limit using a pared Pascal's triangle approach. Of the possible 1.80×10^{16} ($= 2^{54}$) spectra, 1.29×10^{16} remained within the limits of 90 dB dynamic range. Thus, 72 percent of the possible spectral descriptions lie within the assumed bounds and less than 1 bit (out of 54) per frame would be saved if spectral descriptions exceeding a dynamic range of 90 dB were eliminated.

Atal and David[1] published a method for applying phase to an LPC synthesizer which they claim improves the naturalness of the reproduced speech. The method is based on a fixed group delay (phase characteristic) for each pitch harmonic of the voiced excitation. This approach appeared to have potential for reducing the buzziness of the voiced excitation and was easily applied to the SEEVOC synthesizer. Tests using the sample group delay given in the article, however, produced no discernible improvement.

An informal study of the tandeming properties of five systems has been performed. The five real-time systems included the Lincoln Markel LPC (2.4 kbps), a Belgard simulation (2.4 kbps), a CVSD simulation, a 16-kbps APC simulation, and the noisy speech version of the SEEVOC (2.4 kbps), all of which are implemented on LDSPs or LDVTs. All possible combinations of system pairs (including self-tandems) were tested using acoustically clean speech as the input. A trained listener, after a few minutes of exposure, recorded opinions of the performance of the system under test. Briefly, the tandems involving the SEEVOC generally behaved largely like the other system of the tandem pair. For example, the SEE-Belgard tandem and the Belgard-SEE tandem performed very much like a nontandemed Belgard vocoder. The self-tandems usually behaved very much like the same system alone. Many of the non-self and non-SEE tandems exhibited more degradation than either system alone. In general, the tandems involving the SEEVOC were observed to perform well.

Two approaches have been pursued in investigating possible implementations of the SEEVOC. The first emphasizes maximal application of CCD-related technology, while the second implements all functions digitally.

The "maximally CCD" implementation of the quiet acoustic environment analyzer is shown in Fig. 1. The custom CCD samples the speech and computes a short-term log spectrum which is coupled to a microprocessor for spectral envelope estimation and coding. All the functions on the proposed custom CCD device, with the possible exception of the magnitude circuit, have been attempted "on chip." Adequate performance of the DFT, window filter, magnitude function, and log A/D converter have yet to be demonstrated. (Tests of the Reticon CCD CZT DFT have shown

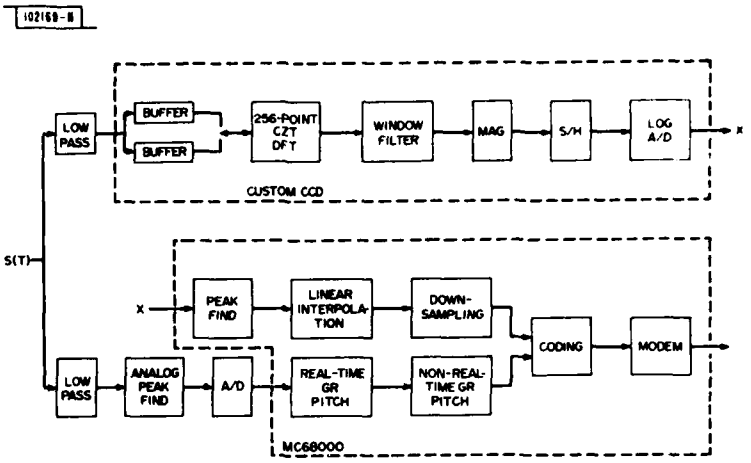

Fig. 1. Quiet environment CCD SEE analyzer.

it to yield inadequate performance in the analyzer.[2]) The remaining functions of the pitch estimation, spectral envelope estimation, coding, and modem interaction should be possible in one or at most two MC68000 microprocessors or the equivalent.

The synthesizer (Fig. 2) requires a real exponential (anti-log), a circuit which has not been attempted on a CCD chip. Performance requirements for the inverse DFT would probably be similar to those of the analysis DFT. The direct convolution operation requires a block which has apparently been built as a smaller isolated device,[3] but its performance is unknown. A slight modification of the shift-and-add structure (Fig. 3) augmented with a complex control structure facilitates pitch synchronous impulse response interpolation simultaneously with the convolution.

At this time, a "maximally CCD" implementation of the SEEVOC appears impractical since a fair degree of untried CCD technology would be involved. The custom synthesizer chip is particularly large and might require partitioning into two chips. The microcomputer portions of the system appear to be within the capabilities of presently available commercial technology.

An all-digital implementation of the SEEVOC appears to be more attractive than the CCD implementation. Analysis of the LDSP-based research version of the system suggests that the maximum allowable butterfly computation time is about 5 μsec for the simplest version of the vocoder. A modified LPCM[4] (equipped with a one-cycle multiply and augmented memory) would support a butterfly time of about 3.75 μsec and therefore might be capable of implementing the vocoder.

An alternative all-digital implementation of the system might involve a GP microprocessor combined with a small, simplified array processor (Fig. 4). A microprocessor such as the MC68000, with its complex interrupt handling capabilities, could handle "real-time" functions such as the serial I/O, the sampled audio input and output, and pitch extraction as indicated earlier. The "mini-array processor" (Fig. 5) could accept an input audio buffer from the microprocessor, perform the analysis, and return the spectral parameters to the microprocessor. For the synthesis, the array processor could then accept spectral and pitch parameters and return an audio output buffer to the microprocessor. This would remove the heavy signal processing computational load from the microprocessor and simultaneously unburden the high-speed

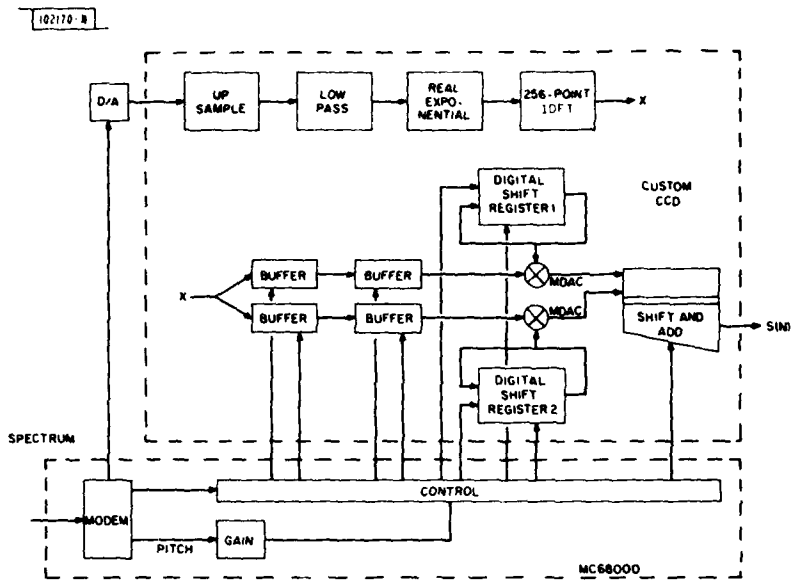

Fig. 2. Zero phase CCD SEE synthesizer.

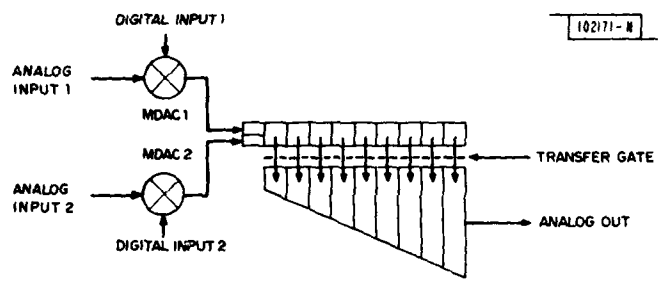

Fig. 3. CCD shift and add convolver.

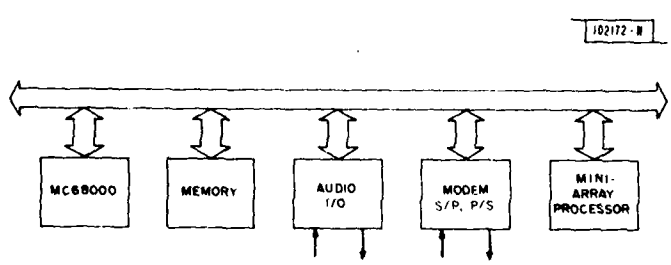

Fig. 4. MC68000-based control structure.

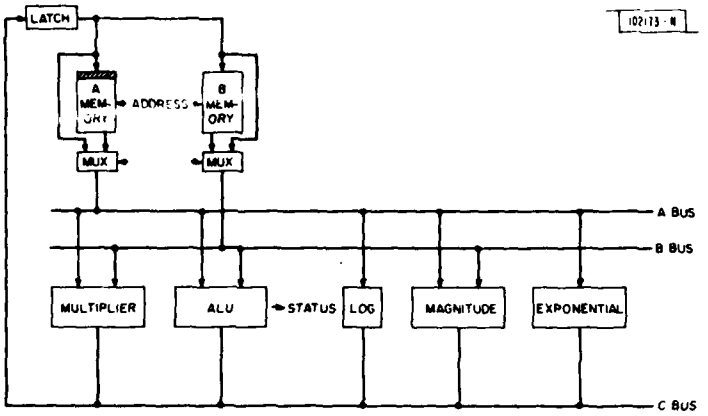

Fig. 5. Example of mini-array processor data structure.

array processor of much complex control. The Motorola/CHI development effort may produce an array processing chip useful in this application.

In conclusion, a SEEVOC implementation based on CCDs to perform any of the major operations does not appear to be practical at this time. The major functions generally have not demonstrated sufficient performance to be useful in a practical system. A digital implementation, however, does appear to be feasible. A modified LPCM or a microprocessor outfitted with a small simplified array processor would be capable of implementing the algorithm in a reasonable amount of hardware. It would also offer flexibility advantages in that minor modifications to the algorithm could be performed relatively easily.

B. Filter-Bank Vocoders

1. High-Quality Channel Vocoder

A "high-quality" channel vocoder operating at two rates (10,150 and 4,900 bps) has been implemented on an LDSP. These rates are a matter of convenience and could be trimmed to the more conventional rates of 9,600 and 4,800 bps. The system is strictly experimental and not real time. The 4,900-bps mode contains a 29-channel spectrum analyzer; each bandpass filter is 120 Hz wide, and its center frequency is 120 Hz removed from its neighbors; the first filter has a center frequency of 120 Hz. The pitch detector is a strict Gold-type as carefully implemented by Blankenship. The synthesizer is also 29 channels with identical filters to the analyzer; spectrum flattening is accomplished by means of a feed-forward AGC on each channel of the first filter bank; the signal is divided by a rectified, low-pass version of itself.

The 10,150-bps mode is derived from the 4,900-bps mode by adding a baseband signal. At the analyzer, the outputs of the first four bandpass filters are added (in phase opposition). The resultant signal is down-sampled by 6:1 and quantized. This quantization is made slightly adaptive by compressing or expanding the quantization grid for each 20-msec frame proportionally to the maximum signal magnitude of that frame. At the synthesizer, this signal is applied (in place of the pitch-derived excitation) as an excitation signal to the first four filters.

2. VLSI Implementation of Filter-Bank Vocoders

Several recent LSI developments have reopened the question of the most suitable implementation for a channel vocoder analyzer and synthesizer. Two papers presented at the 1980 ICASSP[5,6]

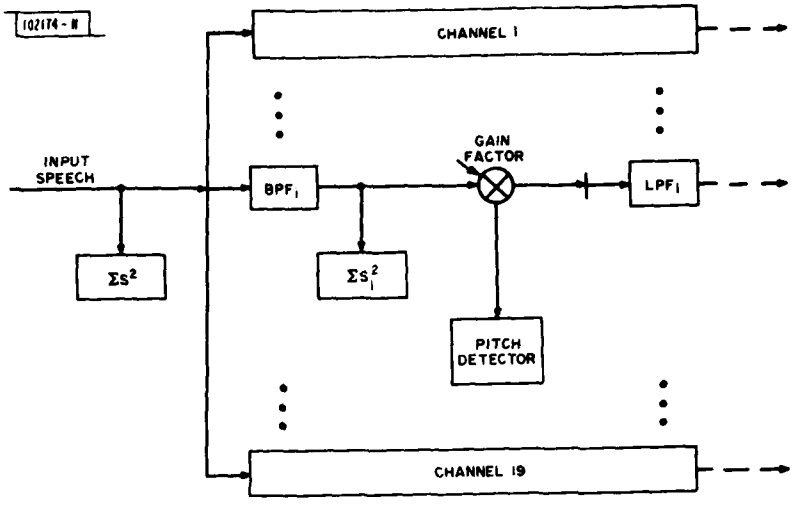

Fig. 6. Energy measurements for estimation of noise in dual-rate channel vocoder.

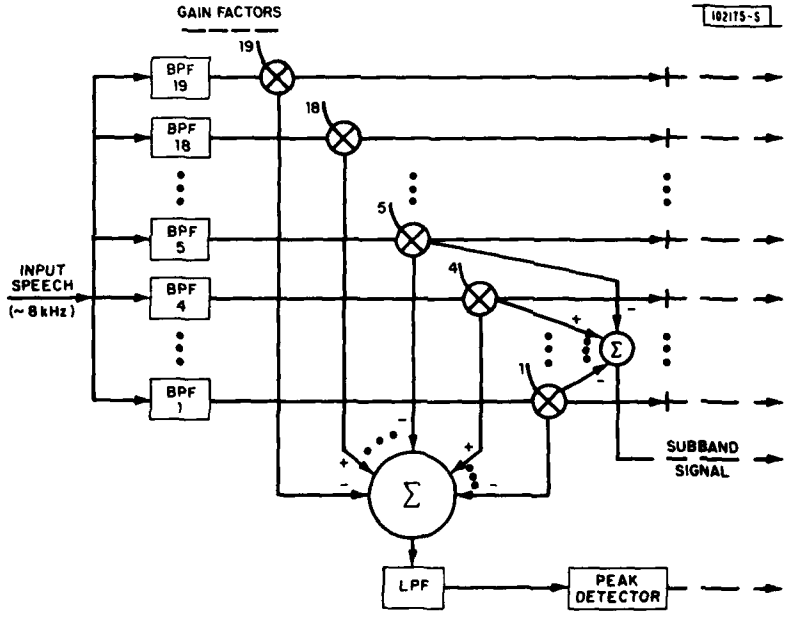

Fig. 7. Dual-rate channel vocoder analyzer with noise suppression.

have described digital signal processing chips that, if available, seem to be applicable to a Belgard-like implementation. A preliminary analysis indicates that two BTL chips[5] are sufficient to perform a Belgard analysis, and that two more are needed for the synthesis (although, in the latter case, the possibility exists that a single chip could be programmed to implement a somewhat simpler synthesizer). The same results apply for the NEC chip;[6] however, this would be a riskier undertaking for attaining acceptable quality because of the shorter register-lengths. At the present time, a more detailed study is being performed to assess the capabilities of both the above chips plus the INTEL 2920 for the channel vocoder. These results will be compared with a parallel study of customized switched-capacitor chips designed to perform the same functions.

C. Dual-R Vocoder with Integrated Noise Stripping

An acoustic-noise suppression feature has been incorporated in the dual-rate channel vocoder previously reported (see pp. 3-6 in Ref. 2). This noise suppression algorithm uses many of the techniques which were developed and implemented to realize a stand-alone prefilter which is discussed in detail in Ref. 7. The inclusion of noise suppression in the vocoder analyzer involves (1) an adaptive noise level threshold marking the probable boundary between noise and noise-plus-speech, (2) a set of adaptively estimated noise components indicating the amount of noise energy present in each of 19 speech spectral bands, and (3) a set of channel attenuations computed as a function of the estimated signal-to-noise ratios.

The noise level threshold is recomputed whenever a frame of speech input is declared unvoiced by the Gold pitch detector. (Fortunately, the types of acoustic-noise backgrounds with which we have been dealing tend to naturally result in an unvoiced decision by the pitch detector.) The total energy in a given frame is computed in double-precision as the sum of the squares of raw speech samples (Fig. 6) and is used to update the noise level threshold. A long time constant ($\approx$3.3 msec) is used to prevent unvoiced segments of speech from increasing the threshold erroneously. (Double-precision multiplies are avoided by using pseudolog and antilog routines.)

Whenever the energy of a frame falls below the noise level threshold, the estimated noise components are updated using the energies computed at the output of each bandpass filter (Fig. 6). Each of the 19 noise levels is recomputed, using a longer time constant ($\approx$200 msec) if the noise energy in that channel is rising, and a shorter time constant ($\approx$90 msec) if the channel energy is falling.

The 19 channel attenuations are updated twice per frame, every 10 msec, using the channel energies computed during the previous 20 msec. The gains are computed as:

$$G(i) = \frac{E(i)\ \text{TOTAL} - E(i)\ \text{NOISE}}{E(i)\ \text{TOTAL}}$$

where

E(i) = energy parameter corresponding to channel i, and

G(i) = estimated ratio of signal to signal-plus-noise.

(Division is done by using the base 2 pseudolog routines and exponentiating the results through a table lookup spanning 40 dB of dynamic range.) The resulting channel gains are then obtained from G(i) through a suppression curve table lookup and smoothed with the previous gains. As shown in Fig. 7, these gains are used to attenuate the output signals of the 19 bandpass filters

before being rectified and also before creating the subband signal. Also, the attenuated outputs are summed to provide a noise suppressed signal for the pitch detector.

A representative set of suppression curves is shown in Fig. 8. (The theory behind the analytical expressions for generating these curves is thoroughly discussed in Ref. 7. The degree of suppression is controlled by a free parameter which is empirically optimized at design time for each environment and fixed thereafter.) The lower horizontal axis is the actual ratio of the estimated signal to signal-plus-noise. The corresponding S/N is shown on the upper horizontal axis. The vertical axis depicts the resulting gain factor to be averaged with the previously computed gain factor for a given channel. The degree of suppression is indicated conceptually as soft, medium, and hard. It is always possible to choose a suppression factor which renders the background noise imperceptible; but when the S/N is low enough, the price paid doing this is the introduction of varying degrees of speech distortion. Alternatively, if a lower suppression factor is used to generate the suppression curve, the speech distortion will be reduced at the expense of readmitting a perceptible level of background noise.

In an attempt to achieve better noise suppression without sacrificing speech quality, further experiments were performed using a suppression curve generated by a very simple empirically determined formula. A closer look at the behavior of the analytically derived suppression curves with respect to a given suppression factor choice reveals that significant attenuation occurs even at relatively healthy S/Ns, and reasonably strong signals were being attenuated as a side effect of suppressing very weak signals; and, conversely, softer suppression to reduce speech distortion guarantees the inclusion of some noise at all S/N values. By using a formula which allows the high, middle, and low points of a curve to be fixed as specified (Fig. 9), a more desirable behavior is obtained: strong signals are never attenuated, very weak signals are always omitted, and the degree of suppression for intermediate S/N values can be varied by redefining the middle point. Although it is not presently known how this new form of suppression strategy impacts intelligibility scores, informal listening experience suggests that overall performance is improved.

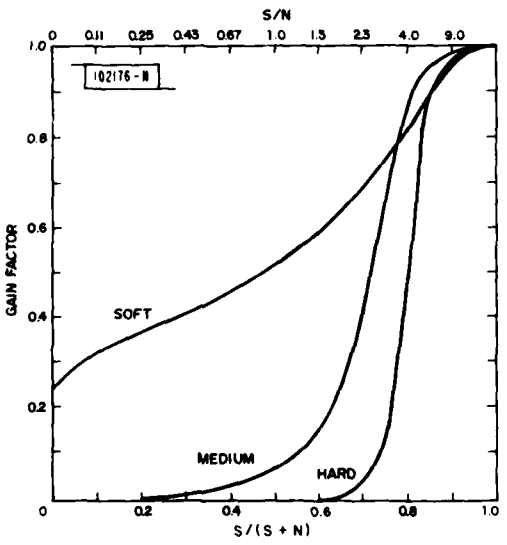

Fig. 8. Example of suppression curves which affect entire range of gains.

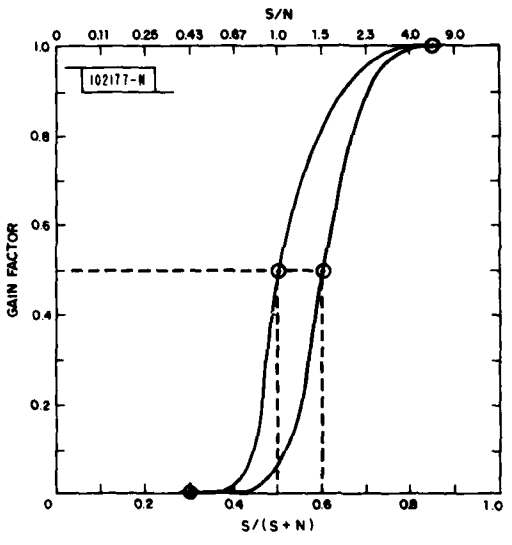

Fig. 9. Example of suppression curves where area of interest is defined.

III. PACKET VOICE TERMINAL AND LOCAL ACCESS AREA

A. Packet Voice Terminal

The present packet voice terminal (PVT) processor design consists of two INTEL 8085 8-bit microprocessor chip sets as shown in Fig. 10. Chip set μC-I interfaces the voice coding device (PCM rates down to vocoder rates) to μC-II, the chip set which runs the control protocol and communicates with the buffer control processor.

The PVT became operational in early January 1980 with μC-II running a simple connect protocol, and formatting data produced by an LDSP programmed to perform a real-time PCM encoding and decoding function (no other speech digitizer was then available). Two sets of this equipment, plus buffer control units and cable modems, made it possible to demonstrate point-to-point packet voice terminal operation over a contention type of cable access area. This demonstration took place in mid-January. Continued work on the PVT added a control pad peripheral device which supported a dial-up capability as well as call-status indicators and bell output response. An INTEL 2910 PCM chip set complex was added to the PVT as an independent speech digitizer, eliminating the need for the LDSP as a speech input device. The PCM chip set addition allows the PVT to function as a complete, free-standing, 64-kbps voice terminal including a dial-up and ringing capability. The overall PCM chip set design is shown in Fig. 11 and emphasizes the penalty paid in additional "glue logic" circuits when adapting the minimal 2-chip CODEC configuration for a specialized requirement. Two of these free-standing 64-kbps terminals were demonstrated for the Wideband Satellite Meeting at Lincoln Laboratory on 24 and 25 March 1980.

A rough estimate of memory requirements for running a Network Voice Protocol, based on an implementation for the PDP-11/45 and assuming a roughly comparable instruction set, yields a need for about 4K of program memory plus approximately 2K of random-access memory (RAM) for tables and buffer space. At present the μC-II chip set is equipped with 2K of read-only program memory (ROM) and 512 RAM locations. However, this can be expanded by installing an additional 4K of ROM and 3K of RAM on the same physical board. If even further memory capacity is called for, a more suitable engineering solution would involve partitioning μC-I and -II onto two separate 7- × 7-in. wirewrap boards, so that each board-chip set will have enough physical space to take advantage of the full 65K memory-addressing capability built into the INTEL 8085 series. A careful reassessment of the present architecture in the light of evolving requirements has culminated in a decision to proceed with a two-board PVT redesign such that μC-II can be augmented with sufficient program ROM and data loadable program RAM to support general-use, such as full-NVP, control of PRN, or ARPANET interfaces, and downloading of executable code from the access area.

However, the above-mentioned redesign of the PVT must also reflect consideration of the following issues:

(1) More time than the 25-μsec margin presently available is needed to support real-time voice protocol operation (packet reordering, sequence number listing, silence handling) at PCM rates.

(2) Efficient connection and communication with Data Encryption Standard (DES) chips is desired.

(3) The necessity for flexible use with Packet Radio and ARPANET interface cards is recognized.

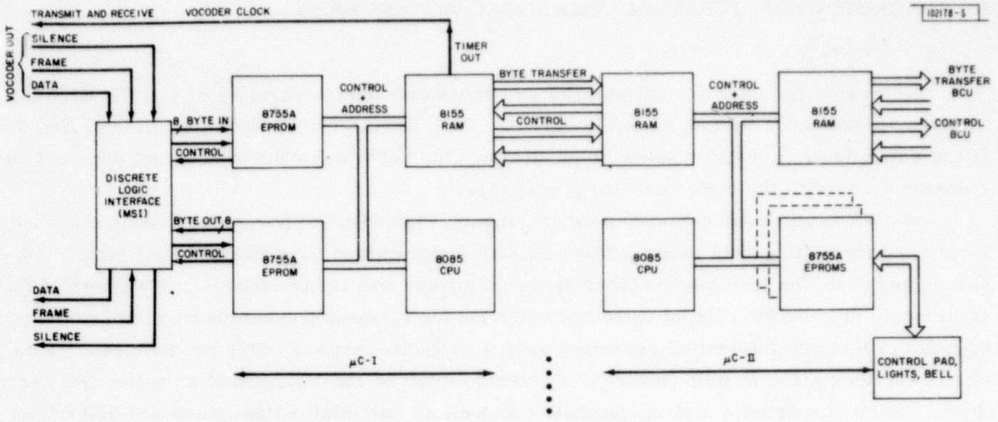

Fig. 10. Present terminal processor.

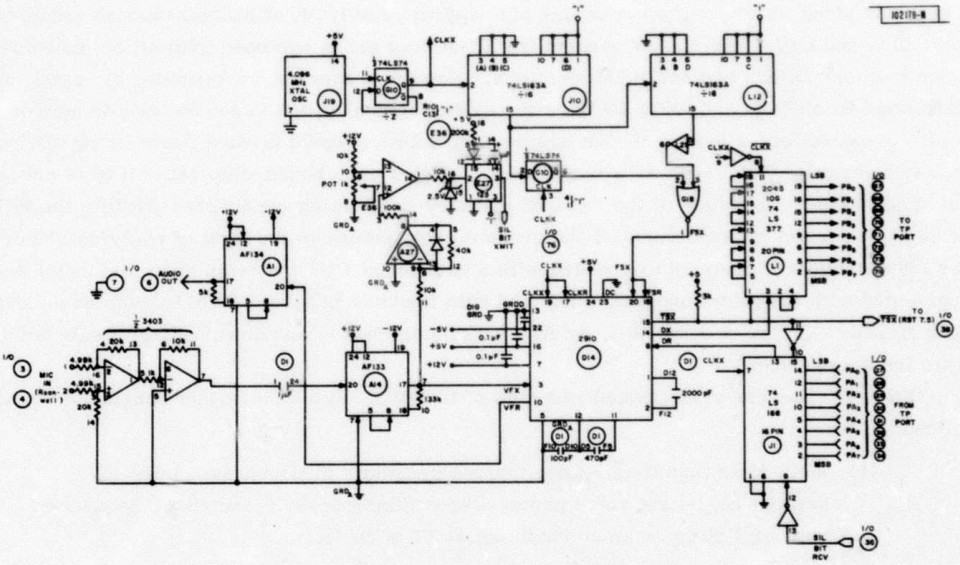

Fig. 11. PCM subsystem.

(4) The design is to accommodate addition of an ASCII serial in-out port to provide a remote access port to the PVT with possible application as a control path for ISI's switched telephone network connection.

(5) Enough memory and computation power is to be provided, enhancing utility of the PVT for other ARPA communication applications.

As a result of considering these more general objectives, a straightforward repartitioning of the PVT into two separate boards for simple memory extension of μC-II is not sufficient and a more global two-board redesign is necessary. The new design will affect μC-II the most and μC-I only slightly in order to satisfy the new requirements. Alterations will include:

In μC-II

(1) Expansion of the 8085 bus to an 8080 bus.

(2) Addition of a type 8237-2 DMA controller chip.

(3) Logic for four independent DMA ports
(a) to BCU, (b) from BCU, (c) to μC-I, (d) from μC-I.

(4) Addition of 4K bytes RAM memory for DMA on 8080 bus.

(5) Addition of 8K program and data RAM on 8085 bus.

(6) Accommodation for Fairchild DES chip set.

(7) Addition of 8251 communication interface for remote ASCII in-out.

In μC-I

(1) Three-bit static switch input for vocoder identification when connected.

(2) An inboard PCM chip set which will be switched into use under software control when no other vocoder is available.

At the present time, it is expected that the core physical configuration will still consist of two 7- × 7-in. Augat boards. Because of the augmentation needed for the revised μC-II, there will only be room for modest additional memory beyond the 12K bytes of RAM mentioned. If future applications require the full 64K-byte memory capacity potentially available, a third card will be necessary connected to the main μC-II bus and loaded only with memory and related support logic. A similar outboard card connected to the μC-II bus will support a DES chip set and interfacing logic. The revised PVT architecture is shown in Figs. 12(a) and (b).

B. Access Area (LEXNET)

1. Buffer Control Processor

Development of firmware for the buffer control processor is in its final stages. The basic framework of the main control program is complete and has been tested. The program controls and monitors DMA transfers to and from both the packet voice terminal (PVT) processor and the modem.

The buffer control program is driven by external events. At start-up, a DMA transfer is enabled from the 2652 serial I/O device to the buffer memory. This enables automatic reception of an incoming packet without CPU intervention. A DMA transfer is also activated from the

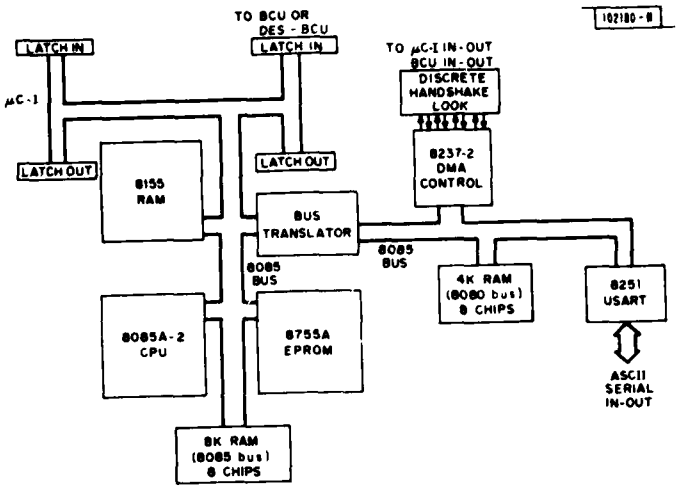

Fig. 12(a). Revised uC-II.

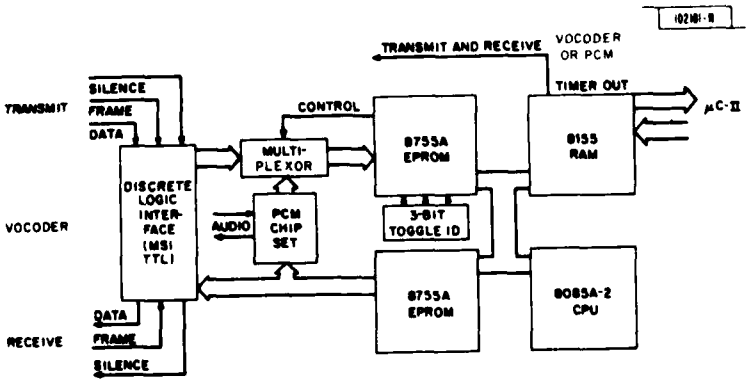

Fig. 12(b). Revised uC-I.

terminal processor to the buffer memory to accommodate a transmit packet. The program then enters a control loop waiting for activity either from the cable or the terminal processor. When a packet is received from the cable, a DMA transfer to the terminal processor is started and a new receive buffer is established. As many as three packets may be queued for transfer to the terminal processor. In addition, the program tests for error conditions on the receive packets. Incomplete packets (probably collisions) are discarded. Complete packets with CRC check errors are transferred to the terminal processor with an error flag.

Only one buffer is provided for transmission. This buffer is loaded by a DMA transfer from the PVT processor. When the complete packet is buffered, the program attempts to transmit the packet according to the collision retransmission protocol. After the packet has been successfully transmitted, the DMA from the PVT processor is again enabled.

Initially, a rudimentary collision resolution algorithm was implemented using a fixed retransmission delay based on the station address. At this time a fully randomized retransmission collision resolution mechanism has been implemented and tested which allows each terminal to make an independent estimate of cable traffic load and adapt its retransmission probability accordingly.

A change in the basic transmission protocol has been made to reduce the latency time of the terminals and involves exploiting the internal address recognition capability of the 2652 to sort the received packets. The 2652 will recognize packets with a particular (8-bit) address or a broadcast address such that a given terminal will ignore packets addressed to other terminals. By using this feature, we can minimize the spacing between packets on the cable. After receipt of a packet, there is a latency interval before the terminal can receive another packet which is related to the time required to set up a new receive buffer. Since it is unlikely that a terminal will receive two packets spaced by less than the latency time, this should have no effect on system performance. If a packet does arrive during the latency interval, for example from a terminal trying to initiate a call when one is already in progress, the receiving terminal will automatically send a signal on the cable which the transmitting station will interpret as a collision. The packet will then be retransmitted according to the random retransmission strategy and will eventually be received after the latency period has elapsed. It is expected that this will occur with very low frequency and will not affect our models of the collision resolution.

The ultimate rate limitation for the current buffer control card design has been investigated. The critical timing path is in the turn-off sequence of the Signetics 2652 communication protocol controller chip. After the last data byte is transferred to the 2652 by the DMA, the CPU has about 14 or 15 bit times in which to assert the end of message (TEOM) control bit. Allowing for the CPU interrupt response time and the possibility of DMA cycle-stealing if the CPU is simultaneously receiving its own transmitted packet (loop-back mode), the fastest cable rate the card can support is about 1.3 Mbps. This rate could be increased either by speeding up the CPU cycle rate from 3 to 5 MHz (and making additional changes on the board to support this rate), or by adding special logic to take care of the setting of the TEOM bit. It has been decided to defer such changes and to operate the system at a cable rate of 1 Mbps, which is sufficient to support initial LEXNET and satellite net packet speech experiments.

A new buffer control card has been designed and built to interface the LEXNET to the mini-concentrator via the UMC-Z80 SIO chip. The design of this modified card is quite similar to the original design. The modified card also uses a Signetics 2652 device to provide a serial data stream to the concentrator via an RS-422 protocol.

2. Trap Control Card

A new version of the trap control card has been constructed featuring three significant improvements. First, the reliability and flexibility of the breakpoint logic have been improved. Also, the card has been equipped to request WAIT states, thus allowing it to work with the faster CPUs (8085A-2). Additionally, a USART and RS-232 data link have been added. This will allow the downloading of executable code from the PDP-11/70, a feature which will be invaluable in the development of software for the PVT.

3. Access Area Modem

A third modem card was built and tested for use in the local access area experiments. Modifications were made on all three modem cards to: (a) delay the receive input to the collision detection circuitry thereby enhancing reliability, and (b) provide circuits to drive indicator lights which signal transmission and collision events on the cable.

C. Packaging and Testing

The prototype LEXNET terminal is housed in an $8\frac{3}{4}$-in.-high standard 19-in. rack sized package (Fig. 13). The basic unit holds up to six 7- × 7-in. wirewrap cards, plus the necessary power supplies. Three of the card slots are committed to the cable modem, buffer control processor, and protocol processor. The remaining three slots are available for a speech processor or for special interfaces. Currently, a 64-kbit PCM speech processor is installed as the fourth card. A front-panel display provides indication of the status of the terminal and activity of the network.

The same basic package design is used for the connection from the LEXNET to the mini-concentrator. It contains three cards: the modem, the buffer control card, and the new concentrator interface card. Two sets of status displays are provided, one for the LEXNET port and one for the concentrator port.

A demonstration program for the voice terminals has been developed. Using a minimal voice protocol, it allows two terminals to communicate 64-kbps PCM speech over 1000 ft of cable. The

Fig. 13. Prototype LEXNET terminal.

simple protocol includes dialing and ringing of the second terminal. A silence detection capability is provided on the PCM card so that packets are transmitted only while the speaker is active. Time stamps are included in the packet header allowing received packets to be played out, with time spacing consistent with the way they were generated.

This program has run successfully with another terminal, programmed as a simple traffic emulator, generating background traffic on the cable.

IV. PACKET VOICE NETWORK PROTOCOL DEVELOPMENT

A second generation of packet voice protocols is being developed by participants in the ARPA Packet Speech and Internet Programs. The protocols will find their first application in the Wideband Satellite Experiments. Two protocols, a Network Voice Protocol (NVP) and a Stream Protocol (ST) on which NVP is supported, are being developed. A number of people and organizations have participated in the development of the protocols, and discussions have taken place at several Network Speech Compression and Internet Meetings. ISI has assumed responsibility for the detailed definition of NVP, and Lincoln Laboratory is doing the same for ST. The first version of a detailed description of ST was described in an Internet Experiment Note (IEN No. 119) in September 1979. A revised version is planned for May 1980. Implementation work on both NVP and ST and miniconcentrator in the packet voice terminal is under way at Lincoln.

The following sections discuss the design goals for the protocols, particularly ST, and give examples of their operation for point-to-point calls and conferences.

A. Protocol Development Goals

Early packet voice experiments made use of a NVP to handle packetization and time sequencing of received packets as well as functions such as agreement of vocoder types associated with setting up point-to-point calls. NVP used the local net protocol (ARPANET) directly to deliver its packets and was independent of and generally incompatible with other protocols in use at the time. Conferencing was handled by a conferencing protocol that was similar to NVP but had additional features to deal with the control requirements of the conferencing situation. Multiple copies of speech packets were sent as required to make a selected speaker's speech available to all listeners.

Since the original NVP made use of the ARPANET protocol directly, extension to other networks, as was done for the Atlantic SATNET, required the creation of a new protocol for each new network. In approaching the design of a second generation of voice protocols, it was decided to start with a more general internet-oriented approach and work toward a protocol that would limit network-dependent aspects to the lowest level. In addition, it was decided to separate protocol functions into two levels. The higher of the two would continue to be named "NVP" and would handle vocoder-type negotiation, ringing, speech packetization, time stamping, etc., as well as dynamic conference control functions. The lower-level protocol, which has come to be named "ST," would provide an internet transport mechanism for the delivery of speech packets for both point-to-point conversations and conferences. The name ST is derived from the word "stream" which refers to the type of traffic load that voice customers offer to a packet network. ST operates at the same level as IP, the DoD standard Internet Protocol, that provides a network-independent transport mechanism for datagram traffic. ST is designed to be compatible with IP and may be viewed as an extension of IP, though it provides a rather different type of service. The two protocols are intended to share the same network ports (i.e., an ST gateway can have the

same local net address as an IP gateway), and the packets for each can be distinguished by the values in a 4-bit field at the front of the packet.

ST differs from IP in being a virtual circuit as opposed to a datagram protocol. In order for a host or gateway to be able to interpret an ST packet header, it must be told about the virtual circuit (called a "connection" in ST) by a setup process that precedes the transmission of any data packets. The setup process involves finding an internet route for the connection and informing all gateways that might have occasion to deal with packets for the connection. During the process, gateways build tables containing information about the connection. It is these tables that permit ST to offer capabilities that cannot be provided by a datagram protocol such as IP. The more important of such capabilities for voice applications are the following:

(1) Efficient forwarding of packet streams – Connection tables allow the use of abbreviated packet headers, resulting in greater efficiency of communication link utilization than can be achieved with datagrams that must carry full network addresses for both source and destination in the packet header. In addition, less processing is required to forward an ST packet since the routing process is carried out at connection setup time and the appropriate next hop for the chosen route (or routes, if alternate routing is desired) is stored in the connection table.

(2) Access to reserved resources – Some networks allow resources to be reserved for use by hosts or groups of hosts. By using such resources, ST can offer improved performance for voice traffic. For example, the wideband satellite network (WB SATNET) provides a stream service that reserves channel bandwidth. Packets using this stream service experience lower average delay and less variance than those using the datagram service also offered by WB SATNET. An ST gateway on WB SATNET can make use of this service by remembering that packets for a particular ST connection are to make use of a particular stream reservation. During the setup process, the ST gateway would negotiate with WB SATNET for the creation of the required stream or the expansion of an existing stream that was to be shared by the new connection.

(3) Traffic control appropriate for voice – Successful packet speech communication requires that network congestion be avoided so that voice packets will arrive in a timely fashion. Congestion can result from node or link failure in the network or from excess load offered by subscribers. With a datagram discipline, the system attempts to recover from congestion by holding off offered packets and/or discarding packets already in the net. This policy causes increased delay and the possible need for retransmission of discarded packets. These effects are not disturbing for most data users, but are very disruptive to voice users. The addition of one more call to a heavily loaded network can cause unsatisfactory performance for many users. ST works to avoid congestion by denying network access to a new call that would result in an overload condition. The connection table information allows the ST agents to predict the average flow requirements of the active connections and to take appropriate action. Such action, in

addition to denying new calls, could include requesting rate changes by variable rate terminals and/or pre-empting lower precedence calls. Gateways carrying a mix of voice and data traffic could also adjust the fraction of capacity allocated to data transmission to keep voice performance in a satisfactory range.

(4) Efficient handling of conference packets – Conferencing requires that all participants receive copies of the voice packets emitted by a conference speaker. Producing multiple copies at the source wastes network bandwidth and is likely to cause local overload conditions for a large conference. Sending a single packet with a multi-addressed header is cumbersome in general, and may be impossible in a large conference when the necessary header size could be larger than the maximum packet size acceptable to the network. ST provides graceful and efficient handling of conference packets by keeping the necessary addressing information in the connection tables and replicating packets only as needed as a packet propagates in the tree-like virtual connection that is provided from the speaker to the listeners. A bit map in the conference packet header is used to control the distribution process.

In order to gain efficiency in networks such as SATNET that have a high per-packet local net overhead, ST provides for a mechanism called "aggregation" which allows that overhead to be shared by packets for different connections which happen to be traveling between a pair of gateways or hosts on that local network. The aggregated packet used in such situations is called an "envelope." At the ST level, it has an envelope header followed by the ST header of the individual packets. These are then followed by the data portions of the packets in the same order. The headers are aggregated together, separately from the data, to allow the header information to be sum-checked independently from the data. In networks such as WB SATNET in which packets can be successfully delivered even though some bit errors have occurred during transmission, the separate aggregation of headers will improve the probability that some or all of the packet contents can be used since only the header portion need be received without errors. Some networks may also allow this portion of the packet to be protected by the use of coding.

The overall goal in the design of the ST protocol has been to do whatever a protocol can do to maximize the probability that voice traffic can be handled satisfactorily in an internetted packet communication system. Obviously, no protocol by itself can guarantee success in this area. Other users of the networks may follow other protocols and cause congestion that results in unsatisfactory voice communications. Success can be achieved only if the networks themselves follow traffic-control policies appropriate for voice applications. A major goal of the Wideband Experiment Program is to explore such control policies and demonstrate their success in supporting packet voice communication on a major scale. ST will contribute to this goal by providing a formalism for interaction between users and traffic-control mechanisms.

B. A Point-to-Point Call Example

This section describes the protocol steps involved in a point-to-point packet voice call. We assume that two speech hosts (voice terminals in the simplest case) named HA and HB are located on two different networks called Net A and Net B. Figure 14 shows a schematic representation of the configuration. The networks are interconnected by a gateway labeled G-AB. The boxes shown

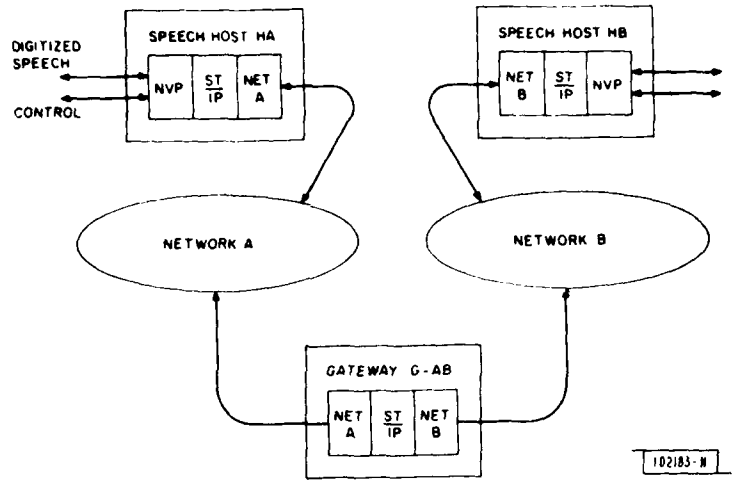

Fig. 14. Configuration for a point-to-point call example.

inside the hosts and gateway refer to protocol processing programs that "understand" the protocols corresponding to the labels on the boxes. Note that the speech host on Net A does not need to have any knowledge about the local protocol that is used in Net B, and vice versa. Note also that the gateway does no processing at the NVP level.

Figure 14 shows IP and ST acting in parallel to support the higher-level NVP. ST is used to deliver speech packets because it can do so more efficiently and satisfactorily than IP. However, the performance advantages of ST do not apply until an end-to-end logical connection has been established. Since connection setup is a relatively expensive process, NVP chooses to carry out preliminary negotiations using IP and asks for an ST connection only when it has determined that call completion is likely.

We will assume that the call in our example originates with a person at HA asking his NVP to set up a call to a person at HB. The protocol does not specify the dialing or other procedure to be used by the person in communicating his desire to the NVP program. The protocol specifies only the rules for exchanging messages across the networks. The first such message from NVP-A to NVP-B is an IP datagram telling NVP-B that the person at A is ready to talk and proposing the use of a particular speech encoder for the call. If HB has compatible encoding equipment and its phone is not already busy, the NVP-B is expected to reply to that effect and indicate that it is ringing the phone. This reply is also carried in an IP datagram.

In order to keep the phone ringing beyond a nominal period (say 3 sec), NVP-A must send a repetition of its original "READY" message. If HB did not have compatible encoding equipment or if its phone was busy, NVP-B would send back a "BYE" message with an appropriate "REASON" code. It could accompany this message with a list of encoder types available to it. NVP-A could select from this list and try again if a compatible combination could be found.

As soon as NVP-A receives an "I-AM-RINGING" message from NVP-B, it initiates the process of setting up an ST connection. The time to set up this connection will overlap with the ringing process and should most often be completed by the time that the phone is answered. If the phone is not answered, the effort of setting up the connection is wasted. If the setup fails because ST can find no route with sufficient capacity, the phone will have rung and possibly been answered

but the call will not be completed satisfactorily. In such a case, the called NVP knows the address (phone number) of the calling party and could provide that information to its user if an appropriate display device were available. A more conservative approach would set up the connection before ringing the phone. A small change to the protocol would allow that variation in the procedure.

The IP datagrams involved in the initial negotiations between NVP-A and NVP-B pass transparently through the gateway between the networks (i.e., the gateway takes no note of the fact that HA and HB are exchanging messages). For the ST connection, however, the gateway must be involved explicitly. To initiate the process, the ST agent at HA sends a "CONNECT" control message to the ST agent in G-AB. Parameters in the message identify NVP-B as the target process for the connection. G-AB makes appropriate table entries and sends a CONNECT message along to the ST agent in HB, which passes it along to NVP-B. NVP-B responds by sending an "ACCEPT" message back to G-AB which, in turn, sends an "ACCEPT" to NVP-A via ST-A. The CONNECT and ACCEPT messages are not simply forwarded by intermediate agents, but are modified by the agents. Each agent is free to specify the identifier to be used for packets to be sent to it on the connection. Thus, the identifier used on the hop between HA and G-AB will generally be different from that used between G-AB and HB. Also, the connection is defined to be full-duplex (i.e., packets can flow in both directions) and different identifiers are used in each direction.

An important parameter on the CONNECT and ACCEPT messages is one called the "FLOW-SPEC." It allows the origin of the CONNECT to indicate what flow rates are needed by the speech encoding equipment for satisfactory communication. Rates are specified by a packet size, a stream interval (time between packets), and a duty cycle (50 percent for speech). If the encoding equipment can operate at more than one rate, the FLOW-SPEC can so indicate, and intermediate agents (gateways) can choose routes to match requests, reporting the result in the FLOW-SPEC returned in the ACCEPT message. The FLOW-SPEC also allows networks to report accumulated estimates of transmission delays and delay variances so that receiving NVPs can make reasonable a priori settings of reconstitution delay parameters used in smoothing played-out speech.

If, in our example, G-AB should determine that the requirements of the FLOW-SPEC for the requested connection could not be met by its path to network B, it would respond to ST-A with a "REFUSE" message instead of sending the CONNECT on to ST-B. NVP-A could abandon the call, seek another route via some other gateway, or perhaps try again with a lower rate encoder if one were available.

Assuming that the connection request was successful (indicated by NVP-A receiving an ACCEPT) and that the phone at HB was answered (indicated by NVPA receiving a READY message for NVP-B), speech packet transmissions could begin. During the conversation, packets are sent only when there is speech activity. Each packet contains a time stamp and a sequence number that are part of the NVP packet header. The time stamp allows the receiver to reconstruct the speech with the proper duration of silence gaps, and the sequence number allows it to detect the occurrence of lost packets.

During the call, the ST agents attempt to provide packet delivery on a timely basis. They do not guarantee that packets will be delivered in the order in which they were sent. They are allowed to drop an occasional packet if congestion threatens. The ST agents do not use retransmission on either an end-to-end or a hop-by-hop basis to provide so-called "reliable" packet delivery service, but they do use an acknowledgment and response protocol for control messages

that attempt to provide a reliable virtual circuit that can be rerouted to get around failed gateways or networks to the extent that viable alternate paths can be found. It is likely that some speech would be lost in the event that a connection had to be rerouted, and in a heavily loaded network it is likely that some of the rerouting attempts would fail. In such a case, the NVPs would receive REFUSE and DISCONNECT messages with appropriate REASON codes.

A call would normally end with one or both NVPs sending "BYE" messages followed by requests to ST to take down the connection. In our example, ST-A (the originating ST agent) would send a DISCONNECT message. ST-B would send a REFUSE with a REASON code indicating that a higher-level protocol had ordered the conversation to be broken.

C. A Conference Call Example

Responsibility for setting up and controlling a voice conference is shared by NVP and ST. ST controls access to the conference and provides a many-to-many connection for each participating NVP that allows it to hand packets to ST to be delivered implicitly to all other participating NVPs and to receive packets from them. Control of the conference "floor," i.e., who is allowed to speak at any given time, is the responsibility of NVP and will not be considered in this example.

In setting up a conference, NVP and ST make use of an entity called an Access Controller (AC) which is a process running on some host computer that can be reached by all participants. The AC may be a public process provided by the internet system and serving many conferences, or it may be a private process serving only a single conference. A conference begins by providing a list of participants to an AC and getting a unique conference name from the AC. If the conference participant list is not known a priori, the AC can be told that the conference is to be open to all who request access or open to any who can give a specified password. In any case, this basic registering of a conference with an AC takes place and the conference name and password, if needed, are distributed to the participants prior to the start of the conference itself.

The conference itself starts with the participating NVPs sending "JOIN" control messages to the AC. If the AC finds the participant's name on its list or finds that an offered password matches, it accepts the participant into the conference and returns a participant number to the NVP. This "PART-NUM" will serve as a shorthand identifier for the duration of the conference. The AC assigns PART-NUMs sequentially (starting from 1) to the participants in the order in which their JOIN commands are received, which may differ from the order of the a priori participant list. If a participant leaves a conference and later rejoins it, he will be given a new PART-NUM. His old number will not be issued to any other participant but will remain unused for the duration of the conference.

The action taken by ST agents in setting up a conference is to open conference connections from participant n to the n-1 participants who have lower PART-NUMs. The ST agent for participant 1 initiates no connections, but waits for the other to connect to it. In order to open a conference connection, an ST agent, either in a host or a gateway, must communicate with the AC to get the internet addresses of the participants to whom connection is desired. Once this information is available, the agent can proceed with the connection. As an example of the procedure, consider the configuration shown in Fig. 15. Assume that participant A was the first to join and that B, C, and D followed in that order. A will attempt connection to no one. B will be told by the AC to connect to A and will send a CONNECT request to G1 which will have to ask the AC about the conference in order to get the address of A, since the CONNECT message merely identified participant 1 as the target. Once G1 has the address of A, it will pass the CONNECT along to A who accepts back through G1 to B.

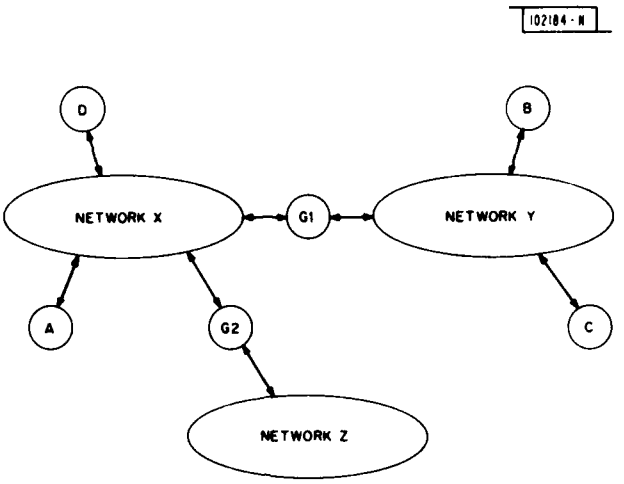

Fig. 15. Configuration for a conference call example.

When C joins, the AC will tell C to connect to A and B. C must send two CONNECT messages, one to B directly and the other to G1 destined for A. G1 may have remembered the address of A and not need to ask the AC for that information, but it is not necessary for G1 to do so since the AC can always provide it again.

When D joins, it will send one CONNECT to A directly and two CONNECTs to G1, one each for B and C. After all these CONNECTs have been accepted, conference packets can be transmitted. When a conference packet is transmitted, the sender puts 1's into a bit map carried in the ST packet header identifying the set of intended recipients. The normal case is for a participant's NVP to send to all other participants. The bit-map position corresponding to a participant is identified by the PART-NUM of the participant. While a participant connects only to those participants with PART-NUMs less than his own, he may accept connections from participants with PART-NUMs greater than his and must maintain a bit map for the entire conference in order to be sure that his packets are going to all other participants.

Let us assume in this example that Net X has broadcast (multi-address) delivery capabilities and that Net Y does not. This means that, when A, D, or G1 are sending packets into Net X, only one instance of each packet is needed. However when B, C, or G1 are sending into Net Y, they must send an instance of each packet to each of the other two. The conference thus represents a heavier load on Net Y than it does on Net X, and the ST agents must take this fact into account as well as the need to replicate packets or not to suit the local network capabilities.

To handle packet forwarding correctly, G1 maintains bit masks that it can apply to the bit maps in arriving packets. For example, if D sends a packet to the conference it will have bits set for A, B, and C. The ST agent at A will see its bit set and pass the packet along to its NVP. G1 will apply its mask for the Net X-to-Net Y direction before inspecting the bits. The mask will allow only the bits for B and C to pass, and G1 will therefore not attempt to forward the packet to A. In more complex configurations the combination of maps and masks prevents packets from looping, a problem that would occur in their absence.

V. MINICONCENTRATOR DEVELOPMENT

The miniconcentrator is a facility being developed by Lincoln Laboratory to support packet speech experiments in the context of the Wideband Experiment Program. The name derives from its function as a concentrator of traffic from many terminals on a local access network (LEXNET) to the wideband satellite network (WB SATNET) acting as a trunk network. It is called a "mini" concentrator because it is designed for a modest data-handling rate that is not intended to fully load the wideband network. It is intended to function as a gateway for both the IP and ST protocols discussed in Sec. IV and as such could serve as a gateway in other network configurations that could include the ARPANET or Packet Radio Nets. In our initial functional design, we have assumed that the voice terminals on LEXNET act as full internet hosts with respect to the IP and ST protocols. However, we allow for the possibility that so-called "dumb" terminals might be in use at some future time that could not handle the full set of protocol interactions. In such a case, the miniconcentrator would be required to supply additional protocol support for those terminals.

The miniconcentrator will also provide a host environment for other processes such as an access controller for conferencing experiments and control, monitoring, and measurement programs.

The following sections first describe the hardware and software design for the system, and then present the current status of the development effort.

A. Hardware Configuration

As shown in Fig. 16, the miniconcentrator hardware consists of a PDP-11/44 minicomputer with UMC-Z80 universal interface computers. The UMC-Z80s are manufactured by Associated Computer Consultants (ACC). As supplied by ACC, each UMC-Z80 has a Z80 microprocessor, three direct memory access (DMA) chips, 20K bytes of random access memory (RAM), some read-only memory with supervisory and debugging firmware, a serial input-output (SIO) chip capable of handling a number of standard serial communication link protocols, and special logic to provide program control of the hardware configuration and to allow the Z80 to directly access PDP-11 memory. In our miniconcentrator design, a UMC-Z80 is required for each network that is connected to the system. The drawing shows two because the first implementation will connect only to the WB SATNET and LEXNET.

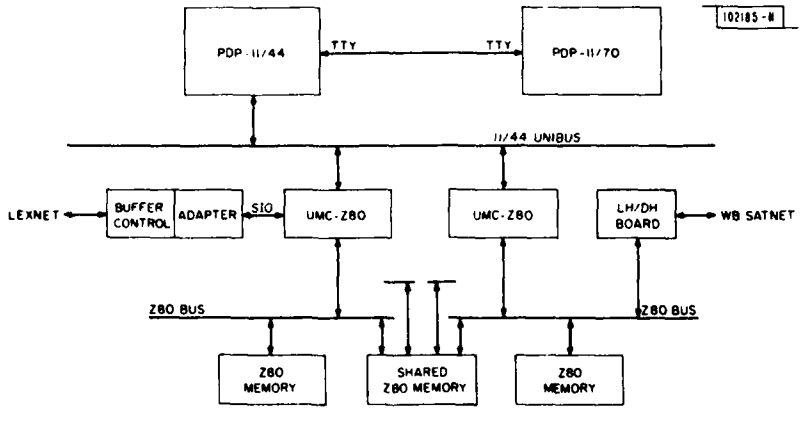

Fig. 16. Miniconcentrator hardware configuration.

Since the miniconcentrator is to be connected to networks with data rates in the megabit-per-second range and the planned experimental program requires access to as much of the network capacity as can be achieved, we have designed the miniconcentrator for maximum packet handling capability consistent with the availability of off-the-shelf components and a schedule that calls for demonstration by the end of FY 80. Since the bulk of the traffic to be handled by the concentrator is traffic to be forwarded through the gateway between the networks, and the data (content) portions of the forwarded packets do not need to be examined by the gateway, we have added an additional memory to the design that can be accessed by the Z80 processors on a multi-port basis. This memory (called Shared Z80 Memory), shown in Fig. 16, is intended to store the data portions of packets to be forwarded so that the load on the PDP-11 UNIBUS and memory will be minimized.

When a packet arrives from a network, it is brought into the memory of the UMC-Z80 connected to that network. Its header is sent to the PDP-11 for processing and its data portion is moved either into shared Z80 memory, if the header indicated that it was a packet to be forwarded, or into PDP-11 memory, if the header indicated that the ultimate destination of the packet was the miniconcentrator itself. When the Z80 sends the header to the PDP-11, it appends pointer and length information so that the packet contents can remain associated with the header. When the PDP-11 schedules a packet for transmission, it sends the header information to the appropriate UMC-Z80 which then picks up the data portion from the location specified by the associated pointer and launches the packet into the network.

Multi-ported memory for microprocessors is not an off-the-shelf item and, consequently, it must be designed and fabricated for this application. Some preliminary design work for the memory has been carried out, but more is needed before construction can be scheduled. It is not likely that shared Z80 memory will be available for the miniconcentrator in FY 80. In its absence, PDP-11 memory will be used for the storage of all packet data. In this mode the maximum data handling rate of the system will be somewhat less than it would be with the shared memory, but the reduction is not expected to be significant for early experiments.

The capabilities of the SIO chip on the UMC-Z80 do not exactly match the requirements of either network. Additional hardware is needed to achieve communication. For the LEXNET, we have built a card which together with a LEXNET buffer control processor card will allow the Z80 to access the LEXNET using one of the standard protocols handled by the SIO chip. BBN is building a similarly motivated piece of hardware to convert the ARPANET very distant host (VDH) protocol used by the WB SATNET into the same standard protocol. In the interim, we are building a card to extend the UMC-Z80 to allow it to handle the ARPANET local-host/distant-host (LH/DH) protocol which is also used by the WB SATNET as a lower-speed alternative to the VDH protocol. The LH/DH card will allow connection to the ARPANET or to Packet Radio nets in other miniconcentrator configurations.

We had originally planned to use our PDP-11/70 computer on a part-time basis to perform the miniconcentrator function. When the PDP-11/44 was announced in the fall of 1979, a relatively inexpensive machine became available with performance in this application very near to that of the 11/70. We decided to order a PDP-11/44 for the miniconcentrator so that we would have full-time availability of a miniconcentrator for experiment use. ISI has ordered another PDP-11/44 and associated UMC-Z80 so that a second miniconcentrator will be available for experiments across the WB SATNET. Delivery of our 11/44 is scheduled for the end of August 1980 and, consequently, that machine will not be available for software checkout or experimental use until just about the end of FY 80. To allow such activities to proceed in the interim, we will use

our 11/45 which is functionally similar but somewhat slower than the 11/44. The 11/45 will be available only a few hours per day for use as a miniconcentrator, but a similar machine at ISI can be used on off-hours (California time). With this arrangement, we expect to have adequate time for software development.

B. Functional Description

Figure 17 is a simplified functional block diagram of the miniconcentrator. Rectangles indicate software modules. Solid arrows show the flow of packets (headers only), and hollow arrows represent the movement of some control information. Other control signals such as communications between the modules and the operating system are not shown. Packets arrive from both networks on the left, and depart to the right. Thus, the UMC-Z80's are shown as split between receive and transmit functions.

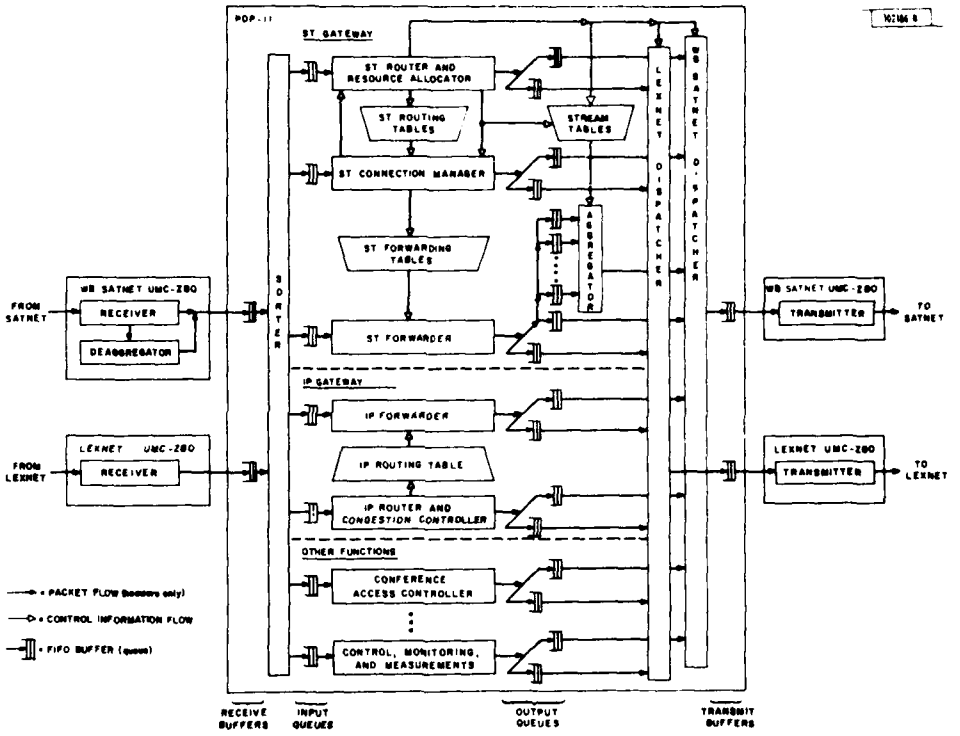

Fig. 17. Miniconcentrator functional block diagram.

The miniconcentrator is a multiprocessor system with the Z80's operating asynchronously with respect to the PDP-11. Since the expected packet load is relatively high (1000 to 2000 packets per second) in relation to the computational speed of the PDP-11, we have chosen not to interrupt the PDP-11 on the arrival of each packet. Instead, interrupts will occur only from a periodic timer set probably for 100 interruptions per second. Some functions will be scheduled on these interrupts, while others will be executed on a lower-priority, almost background basis. The result of this "clocked" operation will be longer average delay through the

system than would be obtained if the processing of each packet was started as soon as possible after its arrival. However, the packet handling rate (throughput) will be higher because fewer CPU cycles will be wasted in switching contexts. The longer average delay will not be significant in relation to the transmission delay in the WB SATNET. While this clocked design appears valid for the SATNET context, it would be less appropriate for use in a many-hop terrestrial network configuration where the accumulated delay could be significant.

Figure 17 shows the functions divided into three groups by horizontal dashed lines. These groups are the ST and IP gateways and "Other Functions," which is an open-ended group of functions that are not currently being implemented. Their description will be left for subsequent reports.

All arriving packets are processed by the RECEIVERs in the UMC-Z80's which separate the packet headers from the data and place the headers in linear receive buffers in PDP-11 memory. Since packets from the WB SATNET may contain aggregated ST packets (see Sec. IV-A), the receiver for the WB SATNET is augmented with a DEAGGREGATOR that breaks these packets apart into individual ST packets before placing the headers in the receive buffers. This function places an extra burden on the UMC-Z80 handling the WB SATNET, but this burden is largely compensated by the reduction in packet rate that results from the use of aggregation.

In order to process the packet headers, the RECEIVERs must check that headers have not been damaged in transmission. The IP and ST protocols provide a sum check of the header for this purpose. If the check fails, the RECEIVER does not pass the header along to the PDP-11. Instead, it records the occurrence of the error and periodically sends a monitor packet to the PDP-11 with accumulated error statistics.

In the PDP-11, the SORTER is awakened periodically to move the headers out from the receive buffers and place them in the appropriate input queues for the gateway and other functions. The bulk of the headers will go into the queues for the ST and IP FORWARDERs, since the other functions deal with control packets that we estimate will make up less than 10 percent of the traffic under reasonable load conditions.

When the SORTER has finished moving the received headers, the FORWARDERs are activated. The IP FORWARDER uses the final destination address in the IP header and information from the ROUTING TABLES to compute the next-hop local net destination and, using that information, builds a local net header appropriate to the chosen net and puts the header on the corresponding output queue. The ST FORWARDER performs a similar function using the shorthand connection identifier in the ST header and information in the ST FORWARDING TABLES to prepare the output header and select the output queue. Because the routing decision process was performed at the time the ST connection was set up, the processing required by the ST FORWARDER in handling a packet is only about one-quarter of that needed by the IP FORWARDER in handling a packet.

If the ST packet is to be sent to the WB SATNET, it is likely that its connection will have been assigned to a SATNET "stream" which represents reserved channel capacity in SATNET. In that case, the FORWARDING TABLE will indicate that the header is to be placed in a particular queue leading to the AGGREGATOR. When it is time to schedule an aggregated stream packet called an "envelope," the AGGREGATOR will be activated (probably every 40 msec). It will use information from the STREAM TABLES to build the envelope header as well as to determine how many packets from the queue can be put into the envelope without exceeding the size limitation that has been negotiated with SATNET. If packets remain in the queue, they may be left until the

next stream interval or moved to the nonaggregated queue, depending upon the particular flow control strategy that is being explored. If the queue becomes exhausted before the maximum envelope size is reached, the AGGREGATOR is free to fill the envelope with other packets for the nonaggregated queue, provided that they have the same WB SATNET next-hop destination.

The DISPATCHERs (including the AGGREGATOR) move headers from the output queues to the transmit buffers where the TRANSMITTERs in the UMC-Z80's pick them up, gather up the corresponding data portions, and launch packets into the nets. When a packet has been successfully transmitted, the TRANSMITTER returns the memory occupied by the data portion to the free storage pool. The DISPATCHERs are activated periodically by the clock (actually before the SORTER, since their timing is more critical). They have the task of metering the flow of packets to the network so that queues do not build up in the transmit buffers in such a way that stream packets would fail to get to the WB SATNET in time for their reserved channel slots.

The ROUTING TABLES used by the IP FORWARDER are made up of a combination of a priori information and information obtained by exchanges of routing control packets with other IP gateways. The IP ROUTER and CONGESTION CONTROLLER (IPR&CC) handles the exchange of routing information with other gateways and takes action when congestion appears imminent. The IPR&CC detects the onset of congestion by observing that one or more of the output queues of the IP FORWARDER have grown to some critical size. The actions allowed by the IP protocol are to send so-called "Quench" messages to the sources of packets that are observed in the queues and to discard packets when storage capacity is exceeded. A source receiving a quench message is expected to halt transmission for a time. Ideally, the quenching process that would be started at a lower threshold would prevent the queue buildup from reaching a level that would cause packets to be discarded. The effectiveness of this and perhaps other strategies will be explored in the course of experimentation with the system.

The ST FORWARDING TABLES are built by the ST CONNECTION MANAGER as a result of the receipt of ST control messages exchanged with other ST agents. When a CONNECT is received, the MANAGER examines the ST ROUTING TABLES to determine the possible routes to the target that might have sufficient uncommitted capacity to support the requirements of the requested connection. In the initial implementation there will be no alternate routes, so the routing process is a go/no-go situation. If the ROUTING TABLES indicate, for example, that a SATNET stream to ISI should be used to reach a voice terminal there, the MANAGER has two tasks to carry out. One is to send a CONNECT message to ISI requesting the continuation of the connection. The other is to ask the ST ROUTER and Resource Allocation (STR&RA) to increase the capacity of the stream to ISI by an amount sufficient to handle the new connection. This amount depends upon the size of the stream prior to the request. If the stream is large enough to be carrying a number of conversations, the TASI advantage can be used and the increment in capacity may be as small as 50 percent of the nominal requirement of the new connection.

The procedures of acquiring allocated capacity and propagating the connection request can go on in parallel, but in that case the MANAGER must be prepared to give back the allocated resource or rescind the connection request as necessary in the event of the failure of either procedure. In the event that both are successful, the MANAGER will make appropriate entries in the FORWARDING TABLES and the STR&RA will adjust the stream size parameter in the STREAM TABLES to correspond to the increased capacity.

In the case of LEXNET, the network does not offer any mechanism for reserving or allocating capacity. All that the STR&RA can do is monitor the average traffic in the network and deny

connection requests that would overload the net and cause excessive delays. Simulation suggests that average traffic loads in the vicinity of 70 percent of the LEXNET transmission rate would be about the maximum useful load.

For networks such as the ARPANET and perhaps Packet Radio nets, it is not possible for an individual host to determine the level of traffic being carried by the net. Therefore, the STR&RA can know only its own traffic to and from the net and would have to exchange control messages with other gateways and hosts to get further information on overall traffic. In the absence of such messages, the STR&RA could use a priori rules such as allowing only one narrow-band connection over its ARPANET port.

The above discussion has presented some simple ways in which the STR&RA, IPR&CC, DISPATCHERs, and AGGREGATOR can interact to attempt to control traffic in the internet environment presented by the Wideband Experiment Program. There are other more complex strategies involving variable-rate speech terminals, embedded coding that would allow packets of low priority to be discarded by gateways, and priority pre-emption of connections that are being considered for experimental evaluation in this environment. The basic gateway structure should be able to support these strategies, but they will not be implemented in the FY 80 time period.

C. Interface Hardware and Software Status

The design of the special I/O board to interface the UMC-Z80 to the ARPANET LH/DH protocol is nearing completion. We have decided to add hardware to the card to reverse bytes in the 16-bit words transmitted to and from the WB SATNET. This byte reversal will cause 16-bit word items in messages to appear in the "natural" form for handling by both the Z80 microprocessors and the PDP-11, avoiding the need for manipulation by software that would reduce overall system packet handling capacity. Also, when the switch is made to the VDH connection to the WB SATNET, there will be no change in the byte order in memory as there would be without the byte reversal.

Upon completion and testing of the special I/O board, it will be replicated for use by the other sites involved in PSAT integration. Parts have been ordered for a total of 8 boards. The wire-wrap board to be used for the I/O card is the special protohex board available from Associated Computer Consultants (ACC) to accommodate mixed IC designs.

To support the checkout of UMC-Z80 interface hardware and the development of software for that device, we have procured the software support system offered by ACC – the hardware manufacturer. In order to operate under the UNIX operating system used on our PDP-11/45, ACC had to produce a new version of the software system. By offering to assist in the debugging process, we were able to move the delivery date of the system significantly ahead. The software system consists of a Z80 assembler, a UNIX driver, and a debugger. Installation of the assembler was completed in January. Although this allowed us to begin familiarizing ourselves with any anomalies of this particular assembler, we needed the driver and debugger in order to run programs. By early February, the driver software arrived and was installed. At the time, little documentation beyond the source code itself was available. Our initial efforts involved running very simple test programs to gain an understanding of the driver's capabilities.

In mid-February the debugger software was delivered. It is the debugger which allows the user to load and execute Z80 programs, examine and modify Z80 memory, and access the Z80 I/O ports. A few problems were encountered in converting the software to run under V7 UNIX, but with assistance from the supplier these have been corrected.

The UMC-Z80 software package does not include any diagnostics. Therefore, once all three pieces of the software system were installed and working, some time was invested in writing code to exercise the processor, the main memory, the memory expansion board, and the DMA (direct memory access) and SIO (serial I/O) chips. The DMA test program uncovered several anomalies in that chip's operation. The applications notes on the DMA chip indicate that the user should be wary of certain functions which may not work exactly as described. Our test program was intended to show which of those warnings applied to the particular DMA chips on our UMC board. In addition, it uncovered areas where the documentation was incorrect or misleading.

The next goal was to try a simple loop-back test by using a DMA channel and one channel of the SIO chip as a transmitter, with another DMA channel and the other SIO channel as a receiver. The SIO chip is a complicated device and, just as in the case of the DMA chip, there are many problems to be aware of when trying to set it up in a particular configuration. Some of these problems are due to modes that do not work as described by the manufacturer (or do not work at all), while others appear to be the result of race conditions at the interface to the DMA chip. Currently, the transmitter appears to be working but the received data are always missing the last byte. We are working with the field service department of the chip manufacturer to determine whether the receiver problems are due to software or hardware errors.

One positive result of all the difficulties we have had with the DMA and SIO chips is that, in the debugging struggle, we have been forced to exercise the chips in nearly all the available modes. We are now fairly well informed on the chips' capabilities and their anomalies, and this should result in more efficient software development in the coming weeks.

D. Miniconcentrator Software Status

After investigating a number of existing and developmental operating systems suitable for supporting the miniconcentrator application, we have chosen the EPOS operating system developed by ISI. EPOS was designed for real-time speech-processing experiments in a PDP-11 environment. It supports virtual memory mapping for user processes with separate instruction and data spaces which will allow the IP and ST gateway functions (see Sec. V-B) to coexist in a single virtual memory and provide other virtual memories for the other functions. EPOS provides a flexible process scheduling mechanism and powerful debugging system. It is quite fast in handling system calls of the type required for our applications. Its handling of terminal I/O is not particularly fast or efficient, but terminal I/O is not an important aspect of operational use of the miniconcentrator.

The software for the miniconcentrator is being written in the C programming language. EPOS was originally designed to support assembly language programs. In order to run C programs under EPOS, it is necessary to provide a library of machine language routines to make EPOS system calls accessible to C programs. ISI has assumed the responsibility of providing this library as well as various other programming and support activities necessary to the needs of the miniconcentrator development effort.

As shown in Fig. 16, there is TTY connection between the miniconcentrator and our 11/70 support computer. Similar connections will exist at other sites. This connection will be used for downloading and debugging software, including the EPOS system itself, and will be used for monitoring and control functions in the operational system. We have written 11/70 programs to support this connection and used them to successfully download and run a version of EPOS configured for our 11/45 by ISI.

While the 11/44 computer will not have any disks, the 11/45 being used in the interim does, and ISI has generated a version of EPOS that can read and write files formatted according to UNIX conventions. Using this file system, we can also transfer software in and out of the EPOS environment either directly by having UNIX read and write the same disk that EPOS uses, or indirectly using the ARPANET file transfer process. The latter technique will allow us to move programs to ISI for checkout in their 11/45.

To date, software work has concentrated on producing a viable support environment and that goal is nearing achievement. Simple C programs have been prepared on the 11/70 under UNIX, compiled and linked with appropriate machine language library routines, transferred to EPOS, and run successfully. Programs have been written to translate the symbol table information generated by the C compiler into a form that can be used by MEND, the EPOS debugger. Full EPOS capability is expected shortly, with completion of the system call library being written by ISI.

In the area of miniconcentrator functional software, the basic inner-loop portions of the IP and ST Forwards, the Aggregator, and Dispatcher (see Fig. 17) were coded and run on the 11/70 for timing purposes at an early stage as a part of the evaluation process that led to the choice of the 11/44. Also, portions of the ST Connection Manager were coded to gain a better understanding of the complexity of the storage structures that would be needed to deal with ST conference connections and to evaluate the ability of C to deal effectively with these structures. However, the bulk of the functional software has yet to be written. We expect that enough will be completed by the end of FY 80 to forward IP packets and to handle ST point-to-point connections. We do not anticipate that conference connections will be supported, at least until the end of the calendar year.

VI. EXPERIMENT DEFINITION AND PLANNING

We have decided to issue the Wideband Experiment Plan and its Supplements as a succession of Lincoln Laboratory Project Reports. This provides a convenient mechanism for review, publication, and distribution control. The original Plan, which appeared as a wideband working note (W-Note-1) in December 1978, has been reissued with minor modifications in Project Report form. Supplement I to the Experiment Plan was issued as W-Note-5 on 10 December 1979 and has been given several small modifications in response to new information, decisions made at the 19 December 1979 wideband meeting discussed below, and other inputs; it has subsequently been issued in Project Report form. The Supplement provides updated information with respect to the three main components of the Experiment Plan, namely the network development plan, the system validation plan, and the advanced systems experiment plan. The updated information includes progress reports and schedule changes on the various items of testbed equipment, further development of system validation plans, and changes and extensions to plans for advanced systems experiments.

Major updated items in the testbed development area include:

(a) Preparation for earth station installation at the four initial sites,

(b) Revisions and more detailed definition of the design and schedule for the Earth Station Interface (ESI),

(c) Definition of protocols for interfacing host processors to the Pluribus Satellite Interface Message Processor (PSAT),

(d) Initial plans for interfacing the Experimental Data Network (EDN) to the wideband satellite network,

(e) Design and initial implementation of a local access network (LEXNET) and packet voice terminal for multi-user packet speech experiments, and

(f) Progress on development of miniconcentrators and voice funnels for speech concentration and internetting applications.

Important extensions cited in system validation planning include a PSAT test plan and definition of a sequence of events for ESI/PSAT integration and host/PSAT integration. In the advanced systems experiment area, major topics updated in the Supplement include:

(a) Definition of site equipment configurations and communication protocols for a series of multi-user packet speech experiments including multiple sites and internetting of the wideband satellite network with the LEXNET and the Packet Radio Network,

(b) More detailed definition of systems-level demand-assignment multiple-access (DAMA) experiments relevant to achieving efficient statistical multiplexing of speech in future military networks and development and testing of software for use in these experiments,

(c) Initial definition of systems-level experiments relevant to future Defense Communication System (DCS) architectures involving routing, systems control, switching/multiplexing, and satellite/terrestrial internetting, and

(d) Description of options for digital voice conferencing experiments, with focus on voice-controlled speaker selection techniques.

Finally, the Supplement includes updated schedules for all parts of the Experiment Plan, as well as discussion of coordination items identified in the original Plan.

Experiment planning activities during FY 80 will be reported in detail in a second Supplement to the Experiment Plan. Below, we report briefly on planning activities focused around Wideband Program meetings held on 19 December 1979 and 24-25 March 1980.

A Wideband Meeting was held at DARPA on 19 December 1979 and was attended by the sponsors and organizational participants in the wideband experiment. The major purposes of the meeting were: to review the Experiment Plan; to discuss experiment objectives, testbed configurations, and equipment acquisition timetables for each of the sites; and to discuss various hardware and software issues which needed to be resolved. The Experiment Plan discussion began with a presentation based on the new Supplement which summarized the updated information described above. Particular emphasis was given to a proposed sequence of three milestone multi-user packet speech experiments and the corresponding equipment configuration requirements at the four sites. The three experiments represent the achievement of successively greater packet speech capability on the wideband system. The milestone dates (September 1980, February 1981, and October 1981) are current estimates of the time by which the sites will have acquired the necessary equipment – earth stations, ESIs, PSATs, and replicas of the LEXNET and voice terminals - to support the experiments. These equipment acquisition and scheduling topics stimulated much discussion at the 19 December meeting as to when and by whom the production of various items would be completed. Final resolution of uncertainties in this area was completed at the 24-25 March Wideband Meeting, which is discussed below.

A Wideband Meeting was held at Lincoln Laboratory on 24 and 25 March and was attended by representatives of the sponsors and all participating organizations. A summary of the meeting

was written and was distributed on 10 April to all attendees. The purposes of the meeting were to deliver progress reports, interchange ideas, establish future plans, and resolve issues that remained in the way of implementation of initial experiment capability. Important examples of the latter were delivery of ESIs, certain Host Access Protocol (HAP) questions, acquisition of miniconcentrators, LEXNETs and voice terminals, and system integration and checkout.

Linkabit's plan is initially to deliver two prototype ESIs, to be followed later by four advanced-development models. The prototypes will go to ISI and Lincoln Laboratory and will be installed in time to support the September 1980 milestone experiment at those two sites. Thus, only ISI and Lincoln will participate in that experiment. All four sites will receive advanced-development models of the ESI in time to support the February 1981 experiments.

The HAP questions requiring immediate resolution were resolved, and certain others were deferred; BBN is documenting them separately. The PDP-11 computers and UMC-Z80 programmable I/O boards for miniconcentrators are to be procured by the individual sites, but Lincoln Laboratory will fabricate the required number of copies of a custom LH/DH I/O adapter board for all sites. Lincoln will jointly produce the basic miniconcentrator gateway software, with cooperation from ISI in the systems area as discussed in Sec. V. Voice terminals to be used for the first milestone experiment (September 1980) will be compatible voice processors already in existence at Lincoln and ISI. Access networks and terminals for subsequent experiments will be produced as required by Lincoln and the other sites.

DoD directives require preparation and approval of an assessment of environmental impact and a statement of potential radiation hazards for any proposed installation of an emitter of significant RF energy under Government sponsorship. Such statements were prepared for the Western Union satellite earth station to be installed at Lincoln Laboratory and were submitted to the Air Force Systems Command via the Electronic Systems Division at Hanscom Field, where Lincoln is located. Final approval of these statements was received on 29 February 1980. The last impediment to erection of the earth station is FCC approval of Western Union's request for a frequency allocation. Such approval is expected in the April time frame. Western Union has stated that construction of the earth station will begin immediately after the approval is obtained and that the earth station will be operational approximately four weeks later.

VII. SATELLITE AND INTERNETTED CONFERENCING

Lincoln Laboratory has provided hardware and software to support voice conferencing experiments as part of the ARPA Atlantic Packet Satellite Experiment and the ARPA Internet Program. Satellite conferencing involving three sites was demonstrated in May 1978 using an early version of the Atlantic Packet Satellite Network (SATNET). A new version of the conferencing control program to work with a new version of SATNET became operational in June 1979. Internetted conferencing between ARPANET and SATNET was demonstrated in September 1979. The capability demonstrated at that time was not robust and suffered from excessive loss of speech due to delay dispersion in the network. Work continued until January 1980 in an attempt to correct these difficulties. Robustness was substantially improved, and a successful demonstration was carried out at the National Telecommunications Conference in Washington, DC on 29 November 1979.

In order to be able to operate from a remote location, a new conferencing control program was written that uses voice energy rather than push buttons to control the conference. The program uses the stream capability of SATNET to carry the speech data, and the datagram

capability to carry control packets that announce the start of transmissions. A controller is allowed to start transmission only when no speech packets have been received in the last 0.4 sec. In the event that two or more controllers start transmitting at about the same time, the stream packets will collide at the satellite and will not be successfully received on their return to earth. The control packets travel in datagram mode and get slower service, but are not subject to collision. The controllers use the arrival of control packets and a precedence structure to resolve the stream collision by aborting transmission from all but the highest-precedence active controller. The voice controlled conferencing program operates successfully, but the long round-trip delay of nearly 5 sec makes it somewhat cumbersome to use.

The long round-trip delay observed during the demonstration was the result of the delay dispersion encountered in SATNET. In an attempt to understand the observed delays, we and BBN (the SATNET contractor) made an extensive series of delay measurements. The statistics suggested a number of areas in which improvements could be made, and BBN made some adjustments to SATNET stream parameters that did improve the delay characteristics somewhat. However, the bulk of the problem seems to lie in the area of communication between SATNET and the gateway host machine in which the conferencing program runs. The software in this area is part of an old version of the gateway program that uses the ELF operating system. Since voice conferencing is the only user of this old ELF gateway, and BBN is not able to support that software at a level sufficient to find and fix the remaining problems, we decided to discontinue our efforts on voice conferencing in SATNET.

VIII. ADVANCED SIGNAL PROCESSOR

An effort is under way to define a programmable digital signal processor capable of five times the throughput of the LDSP, yet consistent with the constraints of reasonable cost and high programmability. Since fundamental logic speeds have increased by only a factor of 2.7 (0.75-nsec gate delay for ECL 100K technology vs 2-nsec gate delay for the ECL 10K logic used for the LDSP), the improvement in throughput must be achieved by a combination of integrated-circuit technology and an efficient architectural structure. Several facets of the design problem are currently being pursued: integrated circuit technology; modular computational subsystems (e.g., multipliers, multi-port memories, etc.); instruction sets; instruction pipelines; architectures; and benchmark programs for evaluating the performance of the machines.

The individual ECL 100K devices are faster than their ECL 10K predecessors. However, they usually come in 600-mil 24-pin packages rather than the 300-mil 16-pin DIP of the 10K series, generally implying that the packages must be spaced farther apart. Wired signal propagation delays, which were significant in ECL 10K-based logic designs, will be an even greater factor now due to both the faster logic and the poorer packing density. For example, a sample 100K wirewrap board of about the same overall dimensions as the LDSP 10K wirewrap boards holds only about 200 vs 500 chips. Therefore, only part of the 2.7 speedup factor will be realizable in a system the size of a small signal-processing machine. However, many of the 100K device types represent a greater scale of integration than ECL 10K and are packaged in an emerging line of 68-pin VLSI chips. Some of these chips are inherently no faster than 10K parts but, since they contain more logic or are configured as more useful functional modules, they may enhance their effective speed by reducing the wiring delays or allowing a more efficient partitioning of the machine.

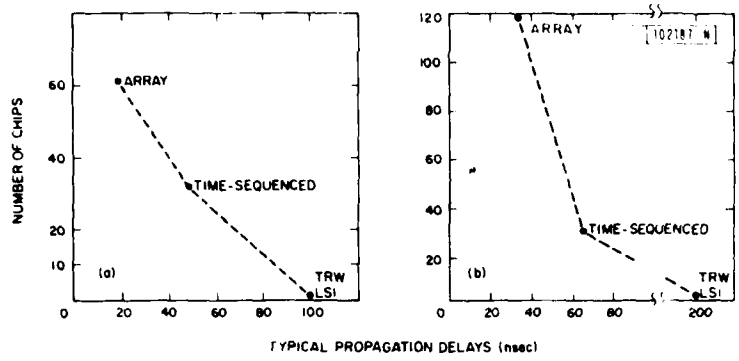

Fig. 18. Multiplier chip count – output delay trade-off.
(a) 16- × 16-bit multiply; (b) 24- × 24-bit multiply.

Several subsystems of the new processor have been investigated including ALUs (arithmetic-logical units), multipliers, large memories, and register files. A 32-bit ALU may be constructed from ten 24-pin F100K 4-bit slice chips, affording a maximum output delay of 12 nsec. A chip-count-vs-output-delay trade-off can be demonstrated in the implementation of the multiplier. Figure 18(a-b) shows the propagation delays (typical, in nanoseconds) and chip counts for 16- × 16-bit and 24- × 24-bit configurations based on an array multiplier, a time-sequenced multiplier operating on four multiplier bits at a time, and a TRW LSI single-chip multiplier. The main candidates for data and program memory are described in Table II. The high-speed, dense (4K × 1) F100470 RAM device may be useful for program memory applications in a machine with a properly matched instruction execution time. Table III indicates chip counts for various register sizes based on a 32 × 9 dual-access stack (Fairchild F100220) and an 8 × 2 multiport RAM (Motorola MECL 10143) aimed at a 10-nsec (typical) dual-read/single-write cycle. Although the F100220 would appear preferable to the 10143 based on chip count considerations, timing specifications and packaging information on this device are still incomplete.

A number of basic instruction sets have been postulated in order to estimate computational power and to demonstrate the programmer's model of the associated architecture(s); a sample follows:

$$M_d \rightarrow r_1 \quad , \quad r_2 \text{ Op } r_3 \rightarrow r_4 \quad , \quad r_5 \rightarrow M_d$$

where M_d represents main data memory, r_i represents a register, and Op represents a typical binary operator. Such a machine could simultaneously perform transfers between main data memory and the register file and support an arithmetic operation between two registers, depositing the results in a third. A pipelined timing sequence for such an instruction might appear as follows:

fetch instruction	decode, compute addresses	$M_d \rightarrow r_1$	r_2 Op $r_3 \rightarrow r_4$	$r_5 \rightarrow M_d$

This instruction set could sustain a high throughput rate but would require the programmer to wait one cycle for certain branch conditions to become visible (e.g., the instruction following a branch would be executed before the branch occurs).

A sample architecture which supports this instruction set and timing arrangement is shown in Fig. 19. This structure has several bottlenecks: data flow at the register file and index

TABLE II ECL RAMS			
Part	Organization	Access Time (Typical) (nsec)	Access Time (Maximum) (nsec)
Fairchild F100470	4K × 1	25.0	35
Fujitsu MB 7071H	[1K × 1 512 × 2 256 × 4]	7.5	10
Fairchild F100422	256 × 4	7.0	10

TABLE III REGISTER FILE CHIP COUNTS FOR DUAL-READ/SINGLE-WRITE CYCLES		
File Size	F100220 Based	MECL 10143 Based
16 × 16	4	24
16 × 24	6	36
32 × 16	4	43
32 × 24	6	65

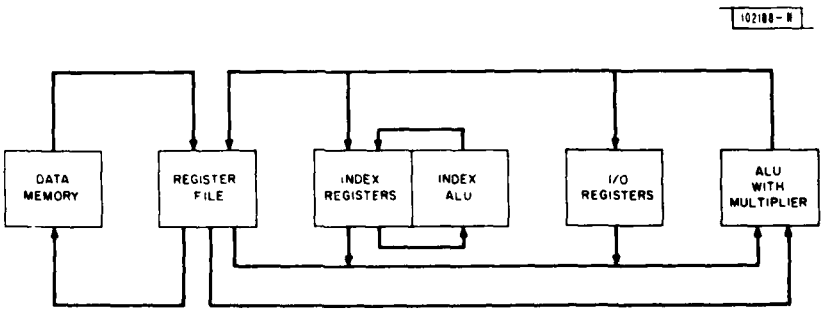

Fig. 19. Architecture of sample machine.

registers, and a slow multiply compared with the speed of the rest of the machine. The exact performance is highly dependent upon the particular parts chosen to implement each functional block. Given a specific choice for the major components used in each functional block, it is possible to pick a specific instruction format and estimate the execution time of each instruction.

Since a large portion of the processing time in many digital signal-processing algorithms is devoted to the execution of orderly, iterative loops, a useful measure of a given processor's throughput may be obtained by examining the execution times for a typical set of these loops. Furthermore, since these loops generally require a small number of program instruction words, this evaluation does not require an inordinate amount of effort. The inner loops of four algorithms important in speech processing were chosen as processor benchmarks: (a) vector dot product (FIR filter, autocorrelation function); (b) lattice filter cell; (c) second-order section; and (d) Radix-2 FFT butterfly. These are depicted in Fig. 20(a-d).

A key result of extensive analyses utilizing these four benchmark-inner loops for a variety of machine architectures is that high processor throughput may be achieved via the combination of a relatively slow (≈10-nsec access), large (≈4K words) main memory with a fast (≈2- to 3-nsec access), small (≈16 word) register file. The register file is particularly effective in that it retains full access to critical data (e.g., filter coefficients, state variables, subtotals, and temporary results), minimizes time-consuming access to main memory, and provides sufficient data flow to keep the ALU operating at high efficiency. Benchmark analysis has also shown that the potential for very high throughput exists through the development of more powerful ALU structures. Some examples of program instructions developed for such structures include the "multiply-add":

$$(R_1 \cdot R_2) + R_3 \rightarrow R_4$$

the "half complex-multiply":

$$(R_1 \cdot R_2) + (R_3 \cdot R_4) \rightarrow R_5$$

and multiple ALU arrangements:

$$R_1 \text{ Op } R_2 \rightarrow R_3 \mid R_4 \text{ Op } R_5 \rightarrow R_6$$

where the R_i's are registers in a register file. One of the greater impediments to realizing high throughput potential derives from data-flow bottlenecks due to limits on main memory access.

In summary, several general classes of machines are currently being investigated, each being evaluated for throughput, programmability, and hardware complexity. At present, both 1 ALU and 2 ALU candidates have been identified which are reasonably easy to program and meet or exceed the target fivefold throughput improvement factor relative to the LDSP. Work is continuing to evaluate the hardware implications of several specific variations of these systems.

IX. CONSORTIUM SUPPORT

Two LDSP complexes were furnished to the Narrowband Speech Consortium test and evaluation facility at AFESD/RADC/EEV, suitably retrofitted for standalone downloading, and modified to support real-time SIO as previously reported. Sets of EPROMS were prepared for SIO versions of the DARPA-sponsored 2.4-kbps SEEVOC and 9.6-kbps dual-rate channel vocoder systems. These algorithms were successfully exercised and documented in all desired test configurations. Field support and maintenance for the LDSP equipments were supplied as needed throughout the several-month duration of the testing.

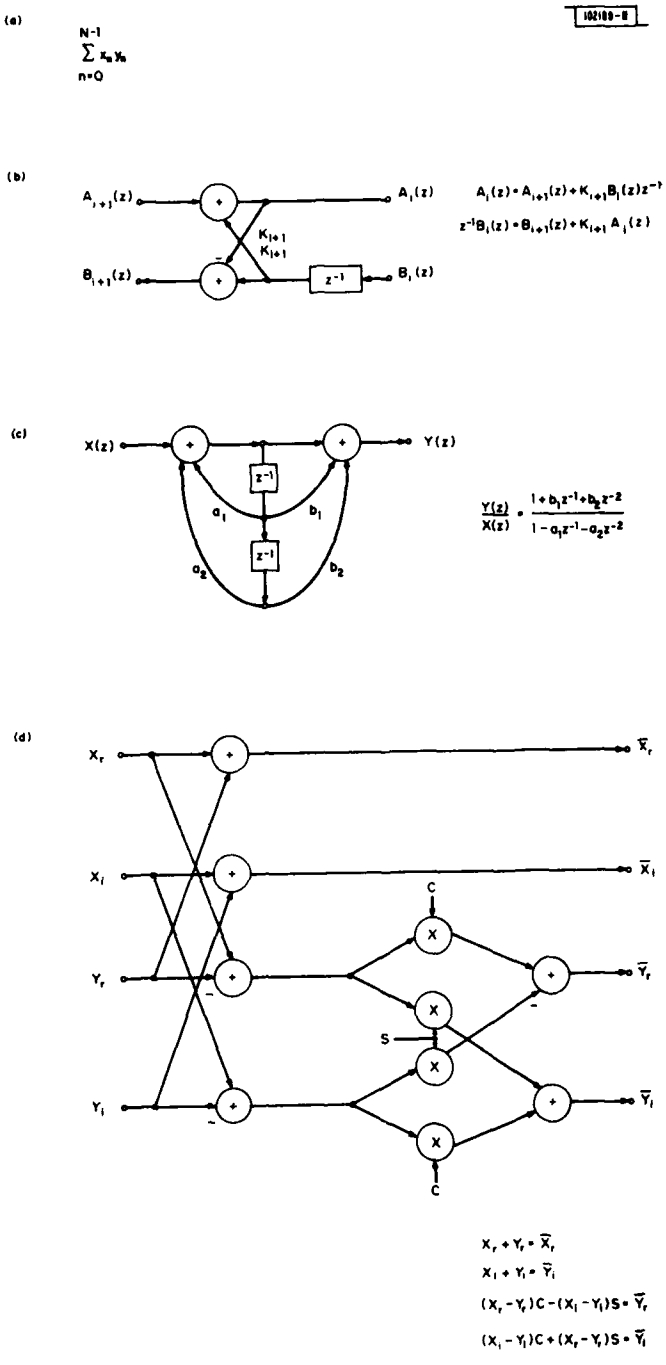

Fig. 20. (a) Dot product (FIR filter, autocorrelation function); (b) lattice filter cell; (c) second-order section; and (d) FFT butterfly.

REFERENCES

1. B. S. Atal and N. David, "On Synthesizing Natural-Sounding Speech by Linear Prediction," Proc. 1979 IEEE International Conference on Acoustics, Speech and Signal Processing, Washington, DC, 2 April 1979.

2. Semiannual Technical Summary, Information Processing Techniques Program, Vol. I: Packet Speech Systems Technology, Lincoln Laboratory, M.I.T. (30 September 1978), pp. 5-8, DDC AD-A066249/4.

3. J. N. Gooding, T. E. Curtis, W. D. Pritchard, and M. A. Rehman, "Programmable Transversal Filter Using CCD Components," Proceedings of CCD '78, San Diego, CA, 26-27 October 1978, pp. 3B-23–3B-30.

4. E. M. Hofstetter, J. Tierney, and O. C. Wheeler, "Microprocessor Realization of a Linear Predictive Vocoder," Technical Note 1976-37, Lincoln Laboratory, M.I.T. (30 September 1976), DDC AD-A033686/7.

5. J. S. Thompson and J. R. Boddie, "An LSI Digital Signal Processor," ICASSP '80 Proceedings, Denver, Colorado, 9 April 1980, p. 383.

6. T. Nishitani, Y. Kawamaki, R. Maruta, and A. Sawai, "LSI Signal Processor Development for Communication Equipment," ICASSP '80 Proceedings, Denver, Colorado, 9 April 1980, p. 386.

7. R. J. McAulay and M. L. Malpass, "Speech Enhancement Using a Soft-Decision Maximum Likelihood Noise Suppression Filter," Technical Note 1979-31, Lincoln Laboratory, M.I.T. (19 June 1979), DDC AD-A074082/9.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER: ESD-TR-80-71	2. GOVT ACCESSION NO.: AD-A092 113
	3. RECIPIENT'S CATALOG NUMBER
4. TITLE *(and Subtitle)*: Information Processing Techniques Program, Volume I, Packet Speech Systems Technology	5. TYPE OF REPORT & PERIOD COVERED: Semiannual Technical Summary, 1 October 1979 – 31 March 1980
	6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s): Clifford J. Weinstein and Peter E. Blankenship	8. CONTRACT OR GRANT NUMBER(s): F19628-80-C-0002
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Lincoln Laboratory, M.I.T., P.O. Box 73, Lexington, MA 02173	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: ARPA Order 3673, Program Element Nos. 61101E/62708E, Project Nos. 0D10/0T10
11. CONTROLLING OFFICE NAME AND ADDRESS: Defense Advanced Research Projects Agency, 1400 Wilson Boulevard, Arlington, VA 22209	12. REPORT DATE: 31 March 1980
	13. NUMBER OF PAGES: 44
14. MONITORING AGENCY NAME & ADDRESS *(if different from Controlling Office)*: Electronic Systems Division, Hanscom AFB, Bedford, MA 01731	15. SECURITY CLASS. *(of this report)*: Unclassified
	15a. DECLASSIFICATION DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT *(of this Report)*

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT *(of the abstract entered in Block 20, if different from Report)*

18. SUPPLEMENTARY NOTES

Supplement to ESD-TR-80-72 (Vol. II)

19. KEY WORDS *(Continue on reverse side if necessary and identify by block number)*

packet speech
network speech
voice conferencing
ARPANET
LEXNET
SATNET
homomorphic vocoding

20. ABSTRACT *(Continue on reverse side if necessary and identify by block number)*

This report describes work performed on the Packet Speech Systems Technology Program sponsored by the Information Processing Techniques Office of the Defense Advanced Research Projects Agency during the period 1 October 1979 through 31 March 1980.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

20765Ø